Technical Report 1008

Unit Performance Assessment System Development

Larry L. Meliza and David W. Bessemer
U.S. Army Research Institute

Seng Chong Tan
University of Central Florida
Institute for Simulation and Training

August 1994

United States Army Research Institute
for the Behavioral and Social Sciences

U.S. ARMY RESEARCH INSTITUTE
FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency Under the Jurisdiction
of the Deputy Chief of Staff for Personnel

EDGAR M. JOHNSON
Director

Technical review by

Mona Crissey
Gene Haga

NOTICES

DISTRIBUTION: Primary distribution of this report has been made by ARI. Please address correspondence concerning distribution of reports to: U.S. Army Research Institute for the Behavioral and Social Sciences, ATTN: PERI-POX, 5001 Eisenhower Ave., Alexandria, Virginia 22333-5600.

FINAL DISPOSITION: This report may be destroyed when it is no longer needed Please do not return it to the U.S. Army Research Institute for the Behavioral and Social Sciences.

NOTE: The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY (Leave blank)

2. REPORT DATE
1994, August

3. REPORT TYPE AND DATES COVERED
Final Apr 90 - Dec 93

4. TITLE AND SUBTITLE
Unit Performance Assessment System Development

5. FUNDING NUMBERS
62785A
790
2114
H1
3204
H1/C7

6. AUTHOR(S)
Meliza, Larry L.; Bessemer, David W. (ARI);
and Tan, Seng C. (U of Central Florida)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Research Institute for the Behavioral and
Social Sciences
ATTN: PERI-IF
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

8. PERFORMING ORGANIZATION REPORT NUMBER
ARI Technical Report
1008

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
--

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
--

11. SUPPLEMENTARY NOTES
--

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release;
distribution is unlimited.

12b. DISTRIBUTION CODE
--

13. ABSTRACT *(Maximum 200 words)*

The U.S. Army Research Institute for the Behavioral and Social Sciences (ARI) developed a prototype personal-computer-based Unit Performance Assessment System (UPAS) to collect network data from simulation networking (SIMNET) exercises to support collective training feedback and research. The UPAS used network data to drive an exercise replay from an overhead view and extracted data for loading a relational database management system tied to data summary graph and table editors. This report describes testing and refinements in the UPAS. We implemented new data filtering systems, created new data types displays, modified data displays, improved user interfaces in response to feedback from trainers. The findings of this research provide input for the design of feedback systems for future applications of distributed interactive simulation (DIS).

14. SUBJECT TERMS
SIMNET
Collective training
Unit performance measurement
Feedback
After Action Review (AAR)
Distributed Interactive
Simulation (DIS)

15. NUMBER OF PAGES
76

16. PRICE CODE
--

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

Technical Report 1008

Unit Performance Assessment System Development

Larry L. Meliza and David W. Bessemer
U.S. Army Research Institute

Seng Chong Tan
University of Central Florida
Institute for Simulation and Training

Simulator Systems Research Unit
Stephen L. Goldberg, Chief

Training Systems Research Division
Jack H. Hiller, Director

U.S. Army Research Institute for the Behavioral and Social Sciences
5001 Eisenhower Avenue, Alexandria, Virginia 22333-5600

Office, Deputy Chief of Staff for Personnel
Department of the Army

August 1994

Army Project Number
2Q162785A790

Human Performance Effectiveness and Simulation

Approved for public release; distribution is unlimited.

FOREWORD

The networking of combat simulators, as illustrated by SIMNET, provides a method for collective training that supplements field exercises. To realize the training potentials of this "electronic battlefield," trainers must be provided with tools to use in identifying and illustrating key exercise events during postexercise After Action Reviews (AARs). This report describes the development of a personal-computer-based Unit Performance Assessment System (UPAS).

Lessons learned from UPAS development are being used to develop procedures for applying the UPAS to AARs after SIMNET exercises at the Fort Knox Combined Arms Tactical Training Center. These lessons are also being used to provide input for the Close Combat Tactical Trainer (CCTT) AAR system, to draft joint service standards for distributed interactive simulation (DIS) feedback and exercise control systems, and to supplement a variety of training technology efforts.

EDGAR M. JOHNSON
Director

UNIT PERFORMANCE ASSESSMENT SYSTEM DEVELOPMENT

EXECUTIVE SUMMARY

Requirement:

To design, test, and refine a personal-computer-based system for measuring unit performance in the simulation networking (SIMNET) environment. The U.S. Army Research Institute for the Behavioral and Social Sciences (ARI) had implemented a prototype Unit Performance Assessment System (UPAS) to collect network data from SIMNET exercises for subsequent analysis of unit performance using an animated replay combined with a relational database to support the preparation of data summary graphs and tables. This early version of the UPAS required enhancements in measurement capabilities to support training research and training feedback through After Action Reviews (AARs).

Procedure:

The capability of the prototype UPAS to collect network data during SIMNET exercises was assessed and improved through software refinements. The capability of the relational database and associated editors to support preparation of graphs and tables was assessed.

Concepts for data displays that might be used in applying Mission Training Plan (MTP) standards for Armor platoons were designed and implemented in the UPAS. The design of data display concepts was guided by the need to integrate terrain, planning, radio communications, and observational data with network data to provide a more complete picture of behavior. New data displays were created, data displays were modified, and user interfaces were modified in response to feedback from trainers at the Fort Knox Mounted Warfare Simulation Training Center (MWSTC).

Findings:

The system for collecting network data was revised to filter unwanted data as well as to prevent loss of critical data. The prototype attempted to collect data based upon Exercise ID, but multiple exercises were often conducted concurrently under the same ID number. We implemented a data filter that limits data collection to entities specified by the user.

We found that the UPAS could not keep up with large data loads causing the loss of irreplaceable data (e.g., data on the firing event that damaged or destroyed a vehicle). We revised the data collection to increase its efficiency and implemented a filter that allows the UPAS to collect irreplaceable data at the expense of other data during peak periods of network activity.

The prototype UPAS had one type of map display, the Plan View, that provided an animated moment-by-moment replay of exercises showing movement and firing events. The map display was enhanced by adding surface terrain features, contour lines, and unit control measures from the unit's operational orders. In addition, three new types of map displays (the Battle Snapshot, Battle Flow, and Fire Fight) were implemented within the UPAS. The Battle Snapshot provided a detailed view of the battle space at a specific second. The Snapshot includes line-of-sight displays, true orientation of vehicles, true orientation of gun tubes, and vehicle IDs. The Battle Flow traced the movement of individual vehicles during the entire exercise or during a critical phase of an exercise. The Fire Fight showed paired firing events and artillery impact areas over a period of time selectable by the user. The Fire Fight was the last map display developed, and it was created in direct response to requests from trainers.

The UPAS system for preparing graphs and tables gives trainers and researchers the flexibility to change graph and table options in response to lessons learned without recourse to formal reprogramming. However, the UPAS graph and table editing systems do place unwarranted restrictions on the graphs and tables that can be prepared.

An Exercise Timeline was developed to show key movement, shooting, and communications events. This timeline can be used to identify points in the exercise that warrant a closer look through the various map displays, and it supports unit performance measurement in its own right.

The application of UPAS data displays to AARs required tools to help trainers select and present displays, because substantial down time results when displays are created during AARs. ARI and IST implemented an AAR Presentation Manager that allows trainers to save static displays (e.g., graphs), add a comment or teaching point to the display, and call up saved displays from a menu during AARs.

Utilization of Findings:

The findings are provided as input for the development of a program of instruction for conducting AARs after SIMNET exercises, and they are provided as input for the development of an

AAR system for the Close Combat Tactical Trainer (CCTT). In addition, they are being used to support a variety of training technology efforts including one on the application of knowledge databases to the preparation of AAR aids. Finally, these findings have been used as input for the development of draft joint service standards for Distributed Interactive Simulation (DIS) requirements for feedback and exercise control systems.

UNIT PERFORMANCE ASSESSMENT SYSTEM DEVELOPMENT

CONTENTS

CONTENTS (Continued)

CONTENTS (Continued)

Page

LIST OF TABLES

LIST OF FIGURES

CONTENTS (Continued)

UNIT PERFORMANCE ASSESSMENT SYSTEM DEVELOPMENT

Introduction

The networking of combat vehicle simulators provides a method for training crews to work together as part of a unit and training units to work together as part of a larger organization (U.S. Army Armor School, 1989a; Thorpe, 1988). Information produced by each simulator, such as its location on the terrain database and the target location of each firing engagement, is broadcast over a network and picked up by other simulators. A computer graphics generator with each simulator is able to reconstruct a realtime "out of the window" picture of the battlefield using broadcast data and data from a common terrain database. The initial application of networked simulators, SIMNET, was developed by the Defense Advanced Research Projects Agency and included simulators for armor and mechanized infantry vehicles (Thorpe, 1988).

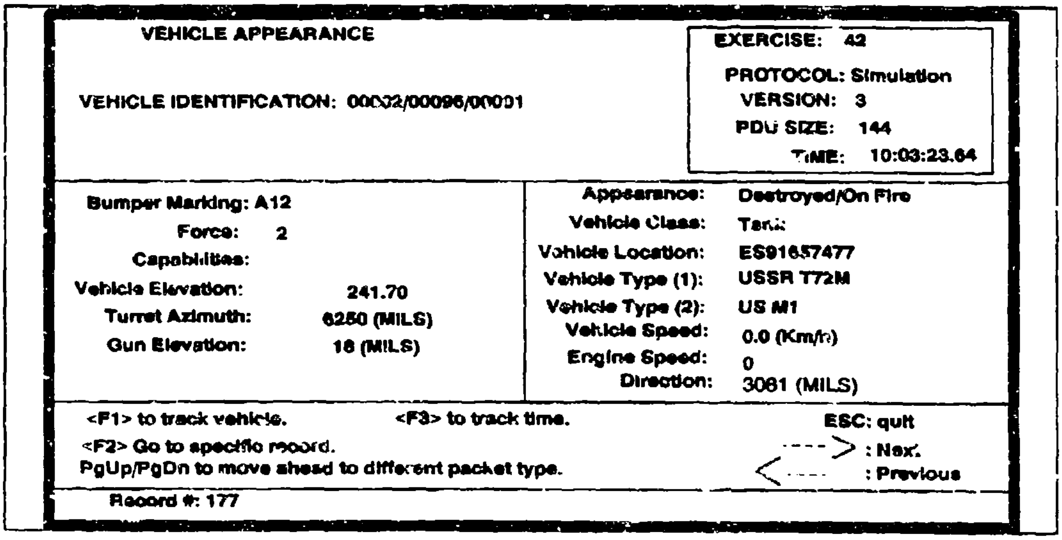
SIMNET includes powerful tools for observing unit performance during and after an exercise. These tools include a "Stealth Vehicle" that allows a trainer or researcher to obtain an "out the window" view of the action from any point on the battlefield and a Plan View Display that allows the action to be observed from a "bird's-eye" view (Thorpe, 1988). However, translating this wealth of available information to a format that supports documentation of training (practice and feedback) and measurement of unit performance is expected to be a substantial task. Further, at sites where SIMNET is used as a trainer, tools to support the collection and analysis of data are not available. Such tools are needed to support training feedback and research on training strategies.

This report describes the design, implementation, and refinement of a personal-computer-based system for measuring unit performance in the SIMNET environment. We prepared this report with three audiences in mind. First, decision makers involved in developing the Close Combat Tactical Trainer (CCTT) require lessons learned from the SIMNET experience that can be applied in developing an After Action Review system for CCTT. Second, training researchers require information about the measurement capabilities within the networked simulator environment. Third, trainers in the current SIMNET environment require information about how to employ the UPAS to support AARs.

The Prototype Unit Performance Assessment System

ARI and Perceptronics initiated development of a low cost, PC-based Unit Performance Assessment System (UPAS) to help collect and analyze data from SIMNET exercises (White, McMeel, and Gross, 1990). This system was intended to support training feedback and research on training strategies for networked simulators, and it focused on data collected from the network.

The prototype UPAS collected six types of protocol data units (PDUs). A Vehicle Appearance PDU (Figure 1) provides information needed to simulate continually each vehicle in a SIMNET exercise. A Vehicle Status PDU is generated by each simulator at fifteen second intervals and contains information on fuel volume, rounds of available ammunition, and odometer readings. A Fire PDU is created by a simulator each time it fires and contains such information as the identify of the target vehicle (if known), type of ammunition used, firing vehicle ID, location of the firing vehicle, and the time of the firing event. An Indirect Fire PDU serves a similar function for indirect fires. A Vehicle or Ground Impact PDU is also generated in association with each firing event showing impact location, engagement range, firing vehicle ID, and target vehicle ID (if a vehicle is hit). Status Change PDUs are generated when there is a major change in the ability of a vehicle to participate in an exercise. These changes include damage or destruction of vehicles due to firing events, administrative kills of vehicles, and administrative reincarnations of destroyed vehicles. The content of each type of PDU is summarized in Table 1.



VEHICLE APPEARANCE

VEHICLE IDENTIFICATION: 00002/00096/00001

EXERCISE: 42
PROTOCOL: Simulation
VERSION: 3
PDU SIZE: 144
TIME: 10:03:23.64

Bumper Marking: A12
Force: 2
Capabilities:
Vehicle Elevation: 241.70
Turret Azimuth: 6250 (MILS)
Gun Elevation: 18 (MILS)

Appearance: Destroyed/On Fire
Vehicle Class: Tank
Vehicle Location: ES91657477
Vehicle Type (1): USSR T72M
Vehicle Type (2): US M1
Vehicle Speed: 0.0 (Km/h)
Engine Speed: 0
Direction: 3081 (MILS)

Record #: 177

Figure 1. Vehicle Appearance Protocol Data Unit.

Table 1.

Data Content of SIMNET Protocol Data Units (PDUs). *

PDU Type	Content
Vehicle Appearance	Vehicle ID Side Vehicle Type Location Status (Alive or Dead) Vehicle Orientation (Heading) Gun Tube Orientation Vehicle and Engine Speed
Vehicle Status	Vehicle ID Fuel and Ammunition Levels Odometer Reading
Fire	Firer ID Ammunition Type Location of Firer Firing Event Number (Numbered Sequentially for Each Vehicle)
Indirect Fire	Firer ID Ammunition Type
Impact	Firer ID Target ID (if Vehicle Hit) Location of Impact Engagement Range Firing Event Number (Numbered Sequentially for Each Vehicle)
Status Change	Nature of Change (Vehicle Damaged, Destroyed, or Reincarnated) Vehicle ID Cause of Change (Usually Direct or Indirect Fire) Vehicle Causing Change (e.g., firer ID)

* Time is included in all PDUs.

The prototype UPAS contained two types of tools to support training feedback and research (White, McMeel, & Gross, 1990), as illustrated in Figure 2. First, the prototype contained a Plan View Display (PVD) to replay the mission or critical segments of the mission from a bird's-eye view (see Figure 3). In addition to showing vehicle location, vehicle orientation, and weapon system -orientation over a grid map, the PVD indicated when each vehicle fired or became a casualty. The UPAS included the capability to magnify the battlefield to the point where the entire display covers an area that is only one kilometer square.

Second, the prototype extracted data from PDUs and loaded these data into a relational database to support the preparation of data summary graphs and tables. To facilitate the application of this database, the UPAS contained menu-based table and graph editors. These editors were used in combination with structured query language (SQL) to create menus of graph and table options.

UPAS loaded data into a relational database management system containing data tables patterned after the National Training Center (NTC) Archive database to support data analysis methods common across the instrumented range and networked simulator training environments (Kerins, Atwood, and Root, 1990). The contents of these UPAS data tables are described in the UPAS Advanced User's Guide (Meliza, Tan, White, Gross, & McMeel, in preparation).

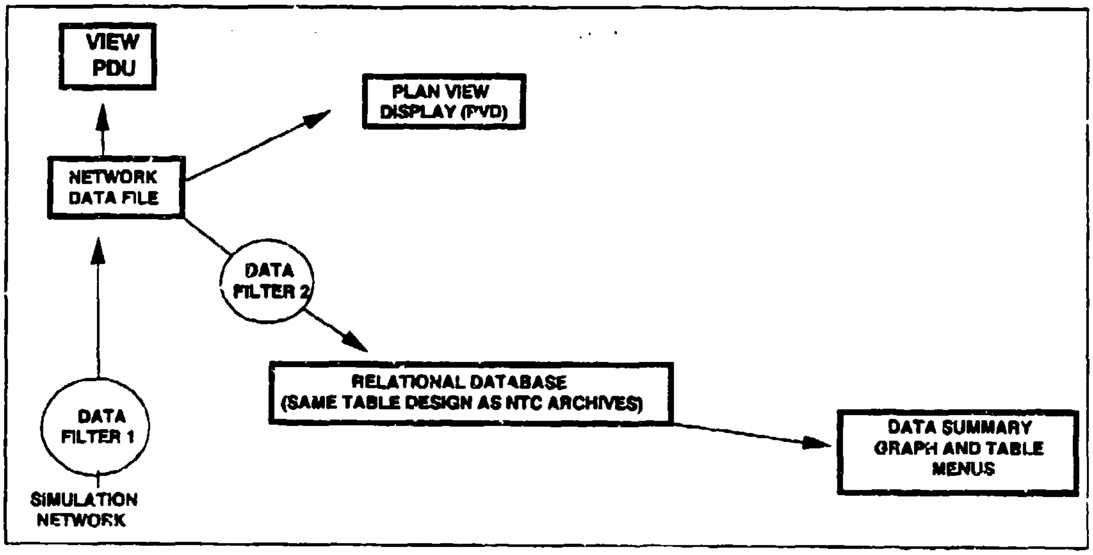


Figure 2. Overview of major components of the prototype UPAS.

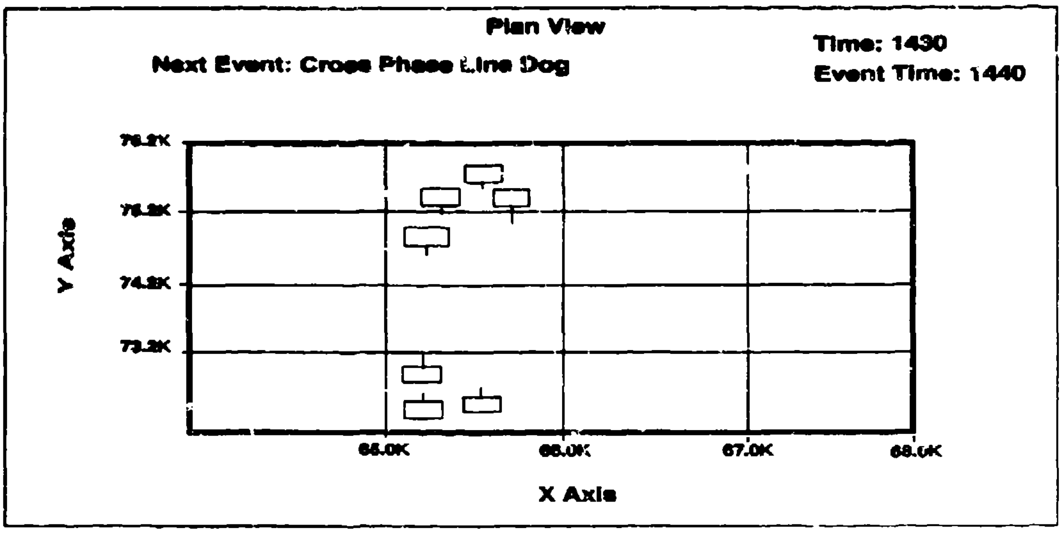


Figure 3. Prototype UPAS Plan View Display.

Goals of the Unit Performance Assessment System Project

The research described in this report was undertaken by ARI and the University of Central Florida Institute for Simulation and Training (IST) to enhance the capabilities of the UPAS as both a research tool and a training feedback tool. The initial objective was to develop data displays that integrate network data with other sources of information to provide a more complete picture of unit performance than is possible with network data alone. Information about the specific mission, enemy, friendly troops, terrain, and time (METT-T) situation under which an exercise is performed is needed to interpret the casualty and position location data collected during a simulated engagement (Kerins, Atwood, and Root,1990; Hiller, 1987; Meliza, 1993). Electronic maps were considered to be a useful tool for integrating network data with terrain data and unit planning data by including major terrain features and unit control measures. Meliza, Bessemer, Burnside, and Shlechter (1992) justified the design of new UPAS data displays, based largely on the need for data displays that might be used to apply standards from Army Training and Evaluation Program (ARTEP) Mission Training Plan (MTP) documents.

The objectives of the UPAS project addressed by the present report are listed below.

1. To upgrade the UPAS to new versions of SIMNET protocols.

2. To test and refine the network data collection and filtering system.

3. To test and refine applications of the UPAS relational database to the preparation of data displays.

4. To test and refine UPAS map displays.

5. To insure that UPAS lessons learned are reflected in the development of future applications of networked simulators.

Upgrade of the UPAS to New Networked Simulator Protocols

Past and Present SIMNET Protocols

Attempts to test UPAS data displays at various points in its development were frustrated by changes in SIMNET protocols and the resulting lack of exercise data for UPAS testing. The prototype UPAS was programmed to collect and analyze data from SIMNET Version 5.0. This version was replaced by Version 6.0 before an adequate amount of 5.0 data could be collected and used to test the UPAS software. Further, the quality of the small amount of 5.0 data available was poor due to technical problems with that version of SIMNET. Examinations of PDU contents showed many instances in which PDUs contained impossible or meaningless values for key parameters such as vehicle locations, fuel levels, and ammunition levels.

The UPAS upgrade to SIMNET Version 6.0 was completed just as Version 6.0 was replaced by 6.61. Once again, limited amounts of 6.0 data were available to support testing of UPAS displays, and the utility of these data were reduced by their poor quality. Meaningful testing of certain UPAS data analysis capabilities could not begin until UPAS was upgraded to SIMNET version 6.61 (Pope and Schaffer, 1991) in February of 1992. The quality of SIMNET exercise data appeared to increase greatly with SIMNET Version 6.61.

The transition from SIMNET Version 5.0 involved changes in the content of PDUs as well as changes in the way data are broadcast over a network. Upgrading UPAS to the newer method of broadcasting data required changing the ETHERNET data capturing hardware and software. This effort involved testing four ETHERNET controller boards ending in the selection of the 3 COM Ethernet Board, based upon cost and the capability to handle network data loads. Adapting to the new PDU protocols required changing (a) the NTC Data Conversion software that loads data into the relational database, (b) the software that allows the user to view the contents of PDUs, and (c) the software that employs PDUs to drive overhead replays of an exercise. This upgrade was accomplished by Perceptronics in association with IST. During the upgrade process, IST was also involved in improving the PVD and implementing new types of overhead displays.

The transition from Version 6.0 to 6.61 involved only a change in the content of PDUs. Once again, modifying the software to support the new PDU contents required changing (a) the NTC Data Conversion software, (b) the software allowing users to view PDUs, and (c) the software that employs PDUs to drive overhead views of replays. New types of overhead displays were upgraded to the new protocols, unlike the previous upgrade.

Distributed Interactive Simulation (DIS) Protocols

Future networked simulators will employ Distributed Interactive Simulation (DIS) protocols (IST, 1991). These protocols are being developed in an ongoing series of Workshops on Standards for the Interoperability of Defense Simulation.

DIS protocols differ from SIMNET 6.61 protocols in terms of broadcast methods, variety of PDUs, and PDU content. To support future DIS applications, the UPAS will need to be upgraded to DIS protocols. IST is currently upgrading the UPAS to DIS Version 2.03 as part of a four service project called Multi-Service Distributed Training Testbed (MDT2). The MDT2 project is sponsored by the Defense Modeling and Simulations Office (DMSO) and involves setting up a testbed devoted to close air support (CAS) operations.

Summary

The UPAS has been upgraded twice already to reflect changes in network protocols. One of these upgrades included hardware and software changes necessary to accommodate a new method of broadcasting data over the net.

We expect that the adoption of standard network protocols by the services and industry will help reduce the number of future upgrades; however, we also expect changes in DIS protocols over time. At present, ARI is preparing to upgrade the UPAS to DIS Version 2.03.

Collection and Filtering of Network Data

More than one exercise can be conducted at the same time over a simulation network. For example, ARI has observed cases at the Fort Knox Combined Arms Tactical Training Center (CATTC) where as many as four exercises were being conducted concurrently with the same control computer. The first problem to be addressed by a network data collection system is limiting data collection to entities involved in a specific exercise. Unless extraneous data are filtered out during data collection, users will face the difficult task of removing these data from the relational database.

A second job of the data collection and filtering system is to address heavy network data loads. Over 90% of the PDUs in a typical SIMNET exercise are Vehicle Appearance PDUs. These PDUs are necessary to simulate the presence of vehicles and other entities on the electronic battlefield. The minimum rate at which these PDUs are generated for each entity is one time per second, and the faster an entity moves, the higher the rate at which it produces these PDUs (Pope and Schaffer, 1991). Data collection systems have a difficult time keeping up with PDU output in large scale exercises involving hundreds of entities.

From a data analysis perspective, the loss of a portion of Vehicle Appearance PDUs is not a serious problem, because the information from one Vehicle Appearance Packet to the next for the same entity shows little change. However, other types of PDUs listed in Table 1 contain data that are irreplaceable. These PDUs include the Vehicle Status PDUs, Fire PDUs, Indirect Fire PDUs, Impact PDUs, and Status Change PDUs.

The specific objectives of this portion of the effort were to

1. design, test, and refine tools for controlling the entities for which network data are collected and
2. design and test tools that will prevent the loss of critical data during peak periods of exercise activity.

The descriptions of data filtering and data loss prevention activities provided below are supplemented with information in Appendix A.

Filtering of Network Data

The initial approach to filtering out extraneous data was to limit the PDUs collected to those with a specified exercise ID number. This approach did not fit the way that exercises are conducted at CATTC, where multiple exercises are often run concurrently under the same Exercise ID number.

The next attempt to filter extraneous data involved limiting data collection to specified entities. This was accomplished by typing in the logical player numbers (LPNs) of all the vehicles involved in a particular exercise. SIMNET uses a three-part entity ID based upon the site, host, and entity number (Pope and Schaffer, 1991). This approach at filtering data was also unsuccessful. One of the first problems encountered was that insufficient information was available at the start of an exercise to set the data filter due to the use of semi-automated forces (SAFOR). Entity IDs for SAFOR are not available before the exercise, and, in certain cases, unplanned SAFOR are generated during the course of an exercise. This problem was addressed by finding out which SAFOR work stations were to be used for a particular exercise (all SAFOR generated from a particular station have the same site and host number) and revising the filtering system to allow for the use of wild cards for SAFOR vehicle Ids (e.g., 2.19.*).

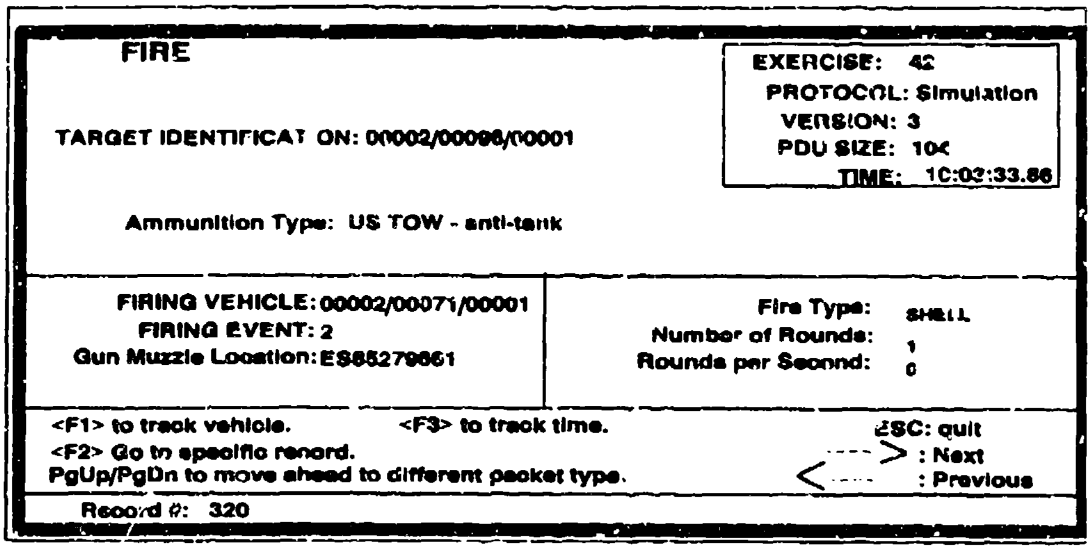
The testing of the filter with the wild card was successful in so far as the data collected was limited to that relevant to a specific exercise. We also found that a slight procedural change was required to collect indirect fire data when the entity filter was employed. To collect indirect fire data the IDs "2.60.*" and "0.0.0" must also be loaded in the filter. The first ID is unique to the CATTC and the second applies across SIMNET sites.

The last remaining problem in removing extraneous data concerns the collection of indirect fire data. The "0.0.0" LPN associated with indirect firing events is not unique to a specific exercise. If indirect fire is being used in other exercises while data are being collected by the UPAS, the UPAS will collect indirect fire PDUs from these other exercises.

Adequacy of Data Collection

There is no ultimate source of information about what constitutes total PDU output during a SIMNET exercise that can be used to assess the percentage of PDUs captured by the UPAS. Instead, we identified three methods for measuring the adequacy of collection of critical data. First, we measured whether the UPAS was collecting all Fire PDUs. We were able to look for missing fire PDUs for specific vehicles, because these PDUs identify the number of the firing event with respect to the firing vehicle. For example, the "2" next to "Firing Event" in Figure 4 means that the vehicle fired one time previously in the exercise. Gaps in the numbered sequence of Fire PDUs mean that some Fire PDUs are not being collected by the UPAS.

Second, we were able to measure the adequacy of the collection of Impact PDUs, because an Impact PDU should be associated with each Fire PDU. In SIMNET, we know where each round lands.



FIRE

EXERCISE: 42
PROTOCOL: Simulation
VERSION: 3
PDU SIZE: 104
TIME: 10:03:33.86

TARGET IDENTIFICATION: 00002/00096/00001

Ammunition Type: US TOW - anti-tank

FIRING VEHICLE: 00002/00071/00001
FIRING EVENT: 2
Gun Muzzle Location: ES65279661

Fire Type: SHELL
Number of Rounds: 1
Rounds per Second: 0

<F1> to track vehicle.
<F3> to track time.
ESC: quit
<F2> Go to specific record.
> : Next
PgUp/PgDn to move ahead to different packet type.
< : Previous

Record #: 320

Figure 4. Example of a Fire Protocol Data Unit.

Third, we were able to check for missing Vehicle Impact and Status Change PDUs by using Vehicle Appearance PDUs to identify vehicles destroyed during an exercise. Once a vehicle is destroyed in a SIMNET exercise, each Vehicle Appearance PDU it generates for the remainder of the exercise states that the vehicle is destroyed and on fire. For each entity generating such a PDU, there should be a Vehicle Impact PDU and a Status Change PDU.

When we applied the above methods for assessing data collection adequacy, we found that the UPAS failed to collect all of the Fire, Impact, and Status Change PDUs. We then made three software modifications to help reduce the loss of critical PDUs. First, we increased the size of the buffer that "holds" PDUs until they are loaded to the hard disk from 20 to 80 kilobytes. Second, we revised the program so that PDUs are bundled into groups of ten and a group, rather than a single PDU, is loaded with each disk access. This change resulted in an estimated five fold increase in the number of PDUs that can be loaded on the hard disk per unit of time. Third, we modified the filter so that the UPAS would stop collecting Vehicle Appearance PDUs selectively when the buffer was more than 80% full.

The results of a test of the new data collection software at IST are summarized in Table 2. Using two computers with equal data collection capability, the old and new software were compared using a seven minute segment of an exercise.

Table 2.

Number of PDUs Collected During a Seven-Minute Exercise Segment by Old and New UPAS Data Collection Software and Percentage of PDUs Falling Into Each PDU Category

	Old		New	
PDU Type:	# of PDUs Collected	% of PDUs	# of PDUs Collected	% of PDUs
Vehicle Appearance	11,518	96.76	21,091	97.25
Impact	4	0.03	5	0.02
Fire	85	0.71	149	0.69
Indirect Fire	23	0.19	40	0.18
Change in Status	1	0.01	1	0.00
Vehicle Status	269	2.26	401	1.85

Comparing the number of PDUs collected between columns 2 and 4 shows that the revised software collected more data per unit of time. Comparing the percentages in columns 3 and 5 shows that the distribution of PDU types remains the same. The latter observation suggests that the benefits of the software changes were mediated by the increased buffer size and grouping of PDUs, and the filter for limiting collection of Vehicle Appearance PDUs was not triggered.

We conducted subsequent testing of the new software at the CATTC facility to find out if all fire, impact, and status change PDUs were collected by the new software. All of the critical PDUs were collected by the UPAS in the data samples we examined; however, the upper limit of network activity that UPAS can handle in large scale exercises remains to be assessed.

We are making further revisions in the UPAS data loading process to increase the capacity to address data loads. This effort involves modifying the UPAS to use variable length records in the raw data file as opposed to the fixed length records currently used. The fixed length record approach requires writing records of fixed size even though some PDUs might be smaller than the record size. It is believed that the variable record length approach will reduce the number of bytes required to write to the disk and improve data collection performance.

UPAS Packet Access Function

The UPAS includes a "Packet Access" function that allows users to examine the contents of PDUs. Figure 4 shows what one of these PDUs looks like when viewed through this UPAS function, and the Advanced UPAS User's Guide describes the Packet Access function (Meliza, Tan, White, Gross, McMeel, in preparation). All of the PDUs are assigned a number by the UPAS as they are collected from the network. The Packet Access provides tools to help a user navigate through the PDUs. The user can navigate through the PDUs by selecting a specific PDU number, vehicle ID, or time. The user can also search for PDUs of a specific type, such as Fire PDUs.

We made extensive use the UPAS Packet Access function in testing the adequacy of data collection, such as when we checked to see whether each Fire PDU was associated with an Impact PDU. Further, we subsequently found the Packet Access function to be useful in examining the PDUs generated by a new version of a computer generated force (CGF).

The Packet Access is a useful tool, but it would be beneficial to add a mechanism for sorting PDUs. For example, it would be useful to select all of the Fire PDUs and then examine these PDUs using the Packet Access function.

Summary

A number of problems were identified in terms of the capability to limit data collection to entities involved in a specific exercises. With one exception, these problems were addressed by modifying UPAS software or the procedures for using the software. The one problem remaining to be addressed is that of limiting the collection of indirect fire data to that relevant to a single exercise.

The original UPAS data collection system could not keep up with network loads during peak periods of activity, resulting in the loss of PDUs. This problem was addressed by increasing the size of the buffer holding PDUs before they are loaded to a hard disk and increasing the rate at which PDUs can be loaded to a hard disk. This problem was addressed also by attempting to filter out less critical PDUs (Vehicle Appearance PDUs) from the data loading process during peak periods of network activity.

Further testing of the software is required in the context of exercises where the data collection capabilities will be stressed with higher network data loads. The goal of these tests is to identify the maximum data load that UPAS can handle without losing critical PDUs. The ideal time to conduct this testing is after the UPAS is revised to load data in a variable length format.

Application of the Relational Database to the Preparation of Data Displays

An important goal of the UPAS project was to create a flexible system that allows data summary options to be changed in response to lessons learned about their utility. The system is considered flexible to the extent that a user can modify the menus without recourse to formal reprogramming of software. By loading data into a relational database, certain problems in flexibility of data analysis were automatically eliminated. The use of structured query language (SQL) allows the user to organize data any way he or she wants.

We refer to loading of network data into a relational database as the NTC data conversion process, because UPAS data tables are patterned after the NTC Archive data tables. Common tables were used to work toward a performance measurement system common to the SIMNET™ and NTC environments. It was initially assumed that utilities developed to analyze performance in one environment might be ported easily to the other environment.

A potential problem in the utility of the database addressed early in prototype development was that of the size of data tables. The high rate at which Vehicle Appearance PDUs are generated tends to produce a glut of information about the appearance of vehicles. This problem was addressed, in part, by giving users the capability to specify the interval at which vehicle appearance data are loaded into the relational database.

To facilitate the application of the relational database to unit performance measurement, graph and table editors were included within the UPAS. Once a new table or graph has been "defined" using these editors, its name is added to the menu of tables or graphs available to all UPAS users. When a graph or table option is selected, the UPAS automatically prepares the table or graph using the exercise files currently being examined.

We foresaw the need to implement an Exercise Timeline to identify when key exercise events occurred, such as when a unit first fires. The Timeline, like graphs and tables, uses data from the relational database. The Timeline, unlike the data summaries created with UPAS graph and table editors, would be fixed. That is, changing the type and manner of data displayed in the Timeline requires reprogramming of software.

The objectives of this portion of the effort were to:

1. find out how much time is required to perform an NTC data conversion as a function of major variables;

2. assess the impacts of data conversion intervals on data analyses;

3. assess the ability of the database, graph editors, and table editors to support flexible data analysis;

4. assess the capability of the UPAS to support common data analysis across SIMNET and NTC environments; and

5. implement, test, and refine an Exercise Timeline.

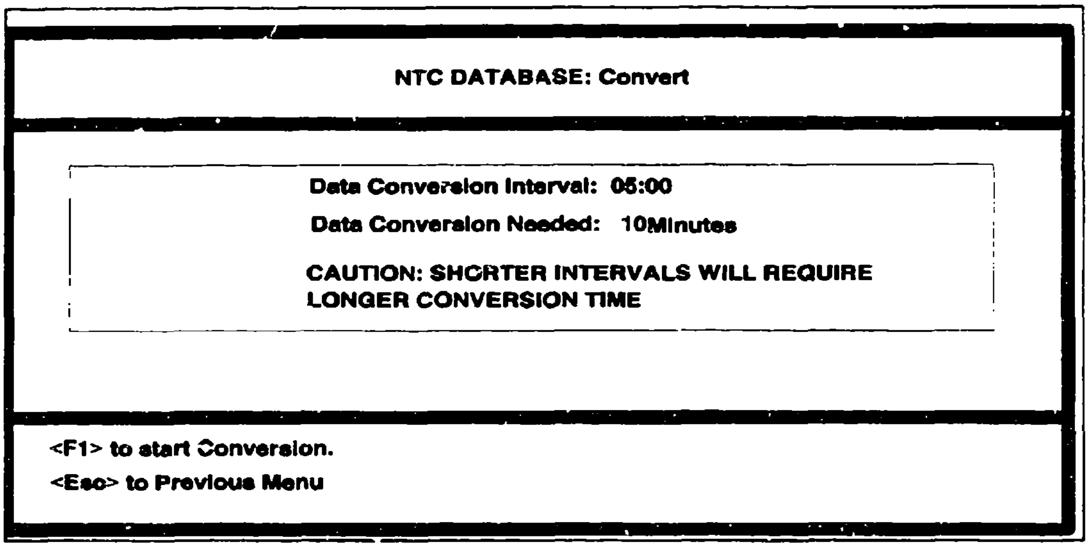
Time Required to Load Data Into a Relational Database

Figure 5 shows a screen for selecting the interval at which location data will be "converted" (i.e., loaded into the relational database). The only table influenced by this change is the ground player location table (GPLT), defined in Table 3, which is based primarily upon data from Vehicle Appearance PDUs and secondarily on data from Vehicle Status PDUs. The data conversion interval determines how often data will be loaded from these PDUs in the absence of any changes in the values of data within the PDUs. For example, if a tank crew were to remain halted for ten minutes and not take any actions that would change Vehicle Appearance or Status PDUs (such as moving the turret or elevating the gun tube) then PDU data would be loaded at the interval specified by the user. If, on the other hand, the Vehicle Appearance or Status PDUs generated by a vehicle change, then the change will cause new data to be loaded into the GPLT table. The default data conversion interval is five minutes.

In our early interactions with trainers at the CATTC facility we were told that data displays should be ready to support AARs within about ten minutes after an exercise. During our continuing interactions with these trainers, they tended to increase the amount of time they would be willing to wait for AAR displays, but thirty minutes appeared to be the maximum amount of time trainers would wait for AAR aids. The time required to load data into this database is a critical factor in determining whether the UPAS can be used to support AARS promptly after an exercise. Therefore it was necessary to assess the factors that influence this data conversion process.

We expected exercise file size and data conversion interval to be major factors in determining the time required to load the database. We examined NTC data conversion times across exercises varying in file size. We hoped to find a strong relationship between file size and conversion time so that we could use file size to estimate the time required to perform data conversions. Each file was converted on a 386-40 computer using the five and one minute data conversion options. The results are shown in Table 4. Although the time required to perform data conversions tended to increase as a function of the size of the exercise file, other variables were clearly at work. We found that the size of the resulting GPLT table was the best predictor of the time required to perform the NTC data conversion (r=0.97).

The size of this table is, in turn, determined by the amount of activity (number of vehicles x length of exercise x amount of action on the part of each vehicle).



NTC DATABASE: Convert

Data Conversion Interval: 05:00

Data Conversion Needed: 10Minutes

CAUTION: SHORTER INTERVALS WILL REQUIRE LONGER CONVERSION TIME

<F1> to start Conversion.

<Esc> to Previous Menu

Figure 5. Screen for selecting intervals at which position location data will be loaded into relational database.

Table 3.

Contents of the SIMNET/NTC Ground Player Location Table.

O Time of Vehicle Status Update
O Player Bumper Number
O Logical Player Number
O Position of Vehicle Expressed in Terms of X-Y-Z Coordinates
O Position of Vehicle Expressed in X-Y Coordinates Relative to the Origin of the Terrain Database
O Vehicle Speed
O Vehicle Direction
O Gun Elevation
O Turret Azimuth
O Engine Speed
O Odomater reading
O Total Amount of Ammunition
O Amount of Fuel Left in Vehicle

Many of the data conversion times shown in Table 4 go well beyond the time available to prepare for AARs. Our only predictor to date of which exercises will take too long to convert is the size of the GPLT table, a variable that cannot be assessed until after data conversion.

We had hopes that porting the UPAS to a faster computer would reduce the time required to perform NTC data conversions. However, data conversion time actually increased with the switch from a 386 computer running at 40 megahertz to a 486 computer running at 50 megahertz or a 486 computer running at 66 megahertz. This was true despite the fact that all standard benchmark tests showed that the 486 computers should be much faster than the 386. It is possible that the problems are due to the engineering of the 486 computers. For example, a more advanced design using a VESA bus, and VESA hard-disk controller with larger cache memory might be able to better utilize the processing speed expected from the 486 computer.

One option to reduce the time required to perform the data conversion was to prepare a version of the software to support AARs that loads all tables except for the GPLT table. This approach would have no effect on the capability of the UPAS to prepare data summaries based upon firing events, but it would reduce the ability of the UPAS relational database to support analyses of unit movement patterns (including the movement portion of an Exercise Timeline) and fuel usage. In this case, research might use the original version of the data conversion software to employ data from the GPLT table. We implemented this option by developing a version of the NTC data conversion software that loads all tables except the GPLT table. Reductions in data conversion times for a sample of exercise files are shown in Table 4.

Another option would be to revise the criteria for loading the GPLT table in a way that reduces the number of events that would trigger the loading of a new row into the table. For example, a significant change in vehicle data might be limited to a change in vehicle location and exclude changes in turret direction or gun tube elevation. This option will not be tested.

Table 4.

Time to Load the UPAS Relational Database and Size of GPLT Table as a Function of Exercise File Size and Data Conversion Interval.

File Size (Megabytes)	Conversion Time: (Minutes:Seconds)		GPLT Table Size (Bytes)	
	5 Minute Interval	1 Minute Interval	5 Minute Interval	1 Minute Interval
4.1	0:32	0:43	23,046	90,942
8.8	5:15	5:53	1,798,692	1,879,836
15.4	2:14	3:02	106,260	431,526
20.9	2:45	3:35	95,082	417,588
25.4	4:28	6:40	276,138	1,128,012
34.4	4:39	6:13	161,046	732,642
35.4	5:17	7:16	217,350	724,638
39.0	6:52	8:42	264,546	923,910
44.7	16:54	19:16	3,832,674	4,620,102
52.5	31:24	34:00	9,584,376	10,174,602
54.4	8:00	12:18	423,660	2,080,488
63.2	9:30	12:32	315,330	1,408,428
68.6	76:03	70:12	20,133,924	20,689,788
80.9	42:38	47:58	10,601,298	11,600,694
86.9	11:48	16:43	419,658	1,997,688
89.2	32:45	36:55	6,134,376	7,640,922

Testing of the UPAS data conversion using update intervals less than one second led us to conclude that when an update interval of less then one minute is selected (e.g., fifteen seconds), UPAS loads all data without filtering. Loading data without filtering, produces very large GPLT tables. In one case, changing the interval from one minute to one second increased the size of the GPLT from 0.5 megabyte to 29 megabytes, many times greater than the RAM capacity of the UPAS. A GPLT created using conversion intervals less than one minute is simply too large to be a useful tool in the PC environment.

Fortunately, many measures of unit performance that require precise position location data might be applied using the map displays described in the next section of this report. These displays use PDUs rather than the relational database.

Other data from the GPLT table that we might want to examine at intervals less than one minute, such as vehicle and engine speed, are not addressed in map displays. Unless data on vehicle and engine speed is loaded into the relational database at frequent intervals, the capability to measure certain aspects of performance is reduced. For example, all vehicles are expected to move at a high rate of speed throughout an assault, and an assault may sometimes last less than one minute. Fortunately, many of the aspects of unit performance that require near continuous updating of data can be trained and evaluated in the context of small unit exercises where network loads are smaller.

Graph and Table Editor Menus

The UPAS graph editor allows the user to: define the layout of a graph by selecting the type of graph (bar or line), write a descriptive name for the graph, title the variables on the horizontal and vertical axes, and specify the types of variables used on the horizontal and vertical axes; write an SQL command that will compute the data needed to display the graph; and prepare menus that will be used to specify the events to be addressed in a particular application of a graph. These menus might be used, for example, to select a specific company and platoon for which firing data are to be displayed.

The UPAS graph and table editor systems impose certain limits on the tables and graphs that can be produced. The tables and graphs are limited to those requiring a single SQL statement. The tables created with the editor cannot contain more than six columns. The graphs created with the graph editor are limited to bar and line graphs, and the variables addressed on the X and Y axes must be time or a simple counting function. These limitations were known early in UPAS development. The extent to which these limitations represented serious drawbacks was assessed as we created graphs and tables in response to recommendations from trainers and researchers.

Most of the graphs and tables of interest to trainers and researchers could be implemented using the UPAS graph and editor systems, but there were patterns of gaps in UPAS capabilities. First, certain tables requested by users could be produced only through the use of multiple SQL statements. Two such tables, a Direct Fire Weapon System Summary and a Fire Support Summary, were produced outside the UPAS Table Editor using Procedural Language. While this language is more complex than using SQL statements alone, it is still far less complex of a task than writing original code in one of the more formal programming languages (such as C++). The code for creating the Direct Fire Weapon System Summary Table is thirteen pages in length, and it is presented in the Advanced UPAS User's Guide (Meliza et al., in preparation). Guidance for using Procedural Language is provided in the XDB User's Manual (XDB, 1990b).

There were also a few cases where desired tables required more than six columns. To meet these requirements it was necessary to split the desired data across two tables rather than consolidating the information in a single table. For situations in which multiple SQL statements are required to create a single table or more than six columns are required it is preferable to use table creation and editing tools that are part of the XDB Relational Database Management System. UPAS contains no tools to add tables to menus of table options when these tables are created outside the table editor.

We discovered a problem in using the graph menu editor the first time we tried to create a graph containing three levels of menus. We found that the UPAS would support only two levels of menus. The editor was reprogrammed to allow three levels of menus to be employed.

Other serious limitations on the graphs are an inability to change the grouping of X values in categories to produce frequency histograms, or to plot percentages or means as a function of a categorical variable. Examples of the types of graphs that cannot be produced include mean number of firing events by unit and miles traveled by unit.

Commonality With the National Training Center Archive Database

One of our goals was to implement a relational database patterned after the NTC Archives relational database to support common unit performance methods across the NTC and SIMNET training environments. More specifically it was hoped that data from one of these environments could be examined using performance measurement tools developed within the other environment. Our first test of this capability involved trying to run SIMNET data using software developed by the ARI Presidio of Monterey (POM) Field Unit to analyze data from the NTC Archive database. We found that it was necessary to perform translations on data in the UPAS relational database to fit data requirements of the POM software. More specifically, the manner in which position location data are reported in SIMNET differs from that in which position location data are reported in the NTC environment. For NTC data, positions are reported in terms of up to six digit coordinates for the x and y axis of a map. For example, the location in the X coordinate might be reported as "22075" and that in the Y coordinate might be reported as "123888." In SIMNET, on the other than, position data are reported using a mixture of characters and numerals. Coordinates are reported using one character and up to five digits. SIMNET maps break terrain into 100 KM by 100 KM blocks with each block being assigned a unique character descriptor in the west/east (X) north/south (Y) planes. Thus SIMNET might report a location in the X coordinate as "R1633" and a position in the Y coordinate as "G8833."

These compatibility problems should disappear with future generations of networked simulators, where the SIMNET map format will be replaced by traditional map formats, like those used at the NTC. For the near term, translating SIMNET location data to the NTC format remains a problem.

Exercise Timeline

To make effective use of UPAS map displays, a user needs to know when critical events occurred during an exercise. The Exercise Timeline is intended to play a dual role in identifying the times of critical events. First, it supports identification of critical events. Second, it shows the time of these events.

An Exercise Timeline is prepared at platoon level using data from the relational database. For a company team level exercise, a separate timeline is prepared for each platoon, the company headquarters (commander and XO), and each attached platoon. In addition to a line displaying the time, the Timeline includes lines for movement, shooting, and communication events (see Figure 6). The movement line shows when a platoon crosses a control measure or comes to a halt. The shoot line shows when a unit first receives direct fire, first delivers direct fire, destroys and enemy vehicle, sustains the loss of a vehicle, or receives indirect fire. The communications line shows message types. An "o" indicates an order, a question mark indicates an information request, "r" indicates a situation or status report, "f" indicates a call for fire, and "m" indicates the message does not fall within any of the other four categories. More detailed descriptions of the algorithms for displaying the events are provided in Appendix B.

The Timeline serves two broad functions. First, it identifies exercise times that might warrant examination during replays. For example, the time when a unit first receives or delivers direct fire might be examined in a replay to find out if the unit was moving in a protective posture when it made contact with enemy. If the user wants to use a replay to examine a unit's use of cover and concealment during halts, the time of each halt is indicated in the Timeline. Second, the Timeline serves as a stand alone tool for examining unit performance.

Unit performance can be examined using each line individually. If few reports are made, the commander should probe for information as indicated by information requests. If a unit is not crossing control measures on time, then a problem in performance has been identified. If that same unit comes to halts frequently, then this might explain the problem. These are but a few examples of how move, shoot, and communicate lines can be used independently to assess performance.. A more complete discussion of Timeline applications is provided in Meliza, Bessemer, et al. (1992).

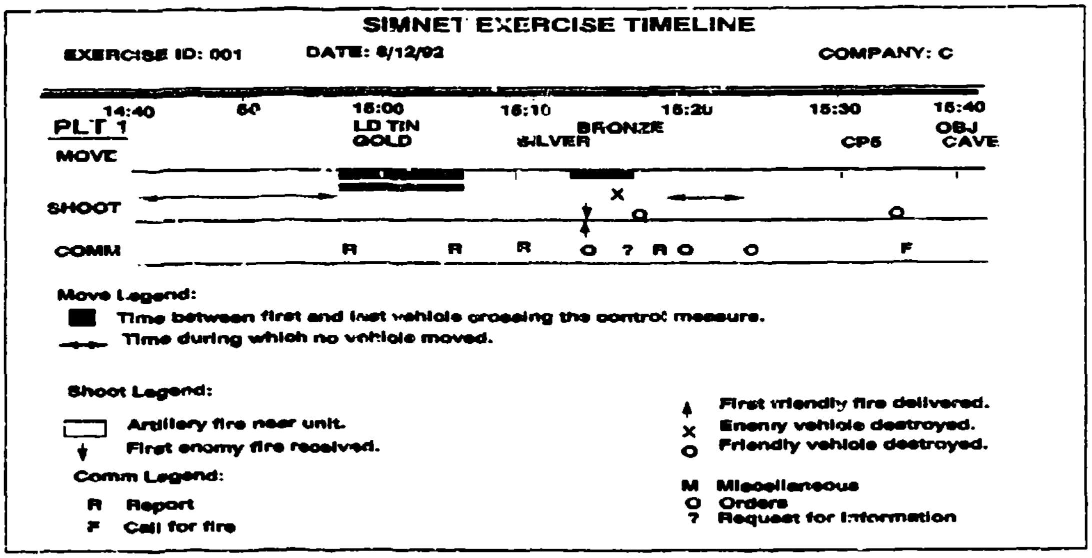


Figure 6. Sample Exercise Timeline.

One of the benefits of the Timeline is its ability to support assessments of the coordination among movement, shooting, and communications. All movement and shoot'ng events in the Timeline should be associated with a report. If a unit is halted when it receives indirect fire, it should report and move out promptly. These are only a few examples of behaviors that can be examined looking across the movement, shooting, and communication lines.

Most the lessons learned regarding the UPAS Exercise Timeline concern the criteria used to decide when particular events have occurred. These lessons are listed below.

1. The time of the first receipt of enemy direct fire was defined initially as the time of the first enemy fire. This variable was redefined as the first firing event resulting in an impact within five hundred meters of any vehicle in the platoon.

2. The initial criterion for displaying artillery impacts was that a round impact within fifty meters of any vehicle in the platoon. This requirement was considered to be too restrictive, because a unit should respond to artillery even if it is beyond fifty meters. In the absence of any hard and fast rule, we changed the criterion to five hundred meters.

3. Halts were initially defined in terms of vehicle speed. A platoon was considered to be halted if all vehicle speeds were zero at the beginning and end of sampled time intervals. In comparing halts displayed on the Timeline with location data, we found discrepancies. Units were sometimes moving at points when the Timeline indicated halts. We changed the definition of halts to reflect changes in location, rather than speed. A platoon is considered to be halted if the locations of individual vehicles change by less than ten meters during an interval. The size of the Timeline interval is equal to the NTC data conversion interval used.

4. The Timeline was also revised in response to comments from trainers at the Fort Knox CATTC. They asked that the displays of initial firing events and casualties be color-coded with the first enemy firing event and casualties inflicted by the enemy displayed using the enemy's colors (Shlechter and Bessemer, in preparation).

5. Finally, we observed that the halt times shown in the Timeline are a function of the data conversion interval used to load the relational database. Using intervals of one minute might show more halts for a unit in comparison with a five minutes data conversion interval.

The UPAS creates Timelines quick enough to support AARS. The time required to generate a Timeline depends upon the size of the GPLT table, as shown in Table 5. These data are based upon the use of a 486-50 computer. It is important to note that the time required to prepare a Timeline is the same for each unit within a particular exercise.

Table 5.

Time Required to Prepare Platoon-Level Timeline as a Function of GPLT Table Size.

GPLT Size (Bytes)	Time Required to Prepare Timeline (Seconds)
61,272	9
417,588	17
1,798,862	45

Summary

The time required to load the relational database will often preclude using this database to support timely AARs. We have prepared a version of the data conversion software that does not load the GPLT table. This approach reduces the amount of time required to load the database but loses the capability to employ certain types of information.

Transferring the UPAS to a faster computer (from a 386-40 to a 486-50) increased rather than reduced the time required to load the relational database. It is possible that 486 computers might need to be engineered differently to take advantage of their speed to reduce data conversion times.

Using data conversion intervals of less than one minute for large exercises results in GPLT tables that are too large to be of practical use for research applications. On the other hand, using data conversion intervals of one minute or longer reduces the ability to use the relational data to apply selected measures of performance requiring a continuous stream of data on vehicle locations, vehicle speed, and engine speed. These measures can be addressed using the relational database in cases where time is not critical (i.e., when data are analyzed outside the scope of a training session) and the size of the data load is small (i.e., for a short platoon level exercise).

The UPAS graph and table editors help to provide a system allowing graphs and tables to be changed or added to menus without formal reprogramming of software. These editors are useful for preparing many, but not all, of the graphs and tables requested by trainers or researchers. The deficiency is due partly to the fact that the editors apply only to those displays that can be created with a single SQL command. The deficiency is mitigated by the fact that multiple graphs and/or tables might provide the same information that the user wants to gather in a single graph or table.

UPAS, on a 486-50 megahertz platform, can generate an Exercise Timeline for a platoon in less than one minute. The Timeline can easily be generated in time to support timely AARs.

The utility of the Timeline can be enhanced by giving the user the capability to select whether enemy or friendly control measures are used when computing control measure crossing times. For example, one way of assessing how well a unit executed a defense is to determine if and when the attacking unit crossed the control measures of the defending unit.

Refinements Common to UPAS Map Displays

The original version of the UPAS contained only one map display, the Plan View. The Plan View provides a replay of an exercise rom an overhead view. Movement and firing events were shown over a grid map lacking terrain features. In addition to adding terrain and control measure data to this display, we saw the need to add two additional types of map displays, a Battle Flow Chart and a Battle Snapshot (Meliza, Bessemer et al., 1992). A Battle Flow was intended to trace a unit's movement over the course of an exercise, and Battle Snapshots were intended to provide a static view of the battlefield at discrete points in time. The Snapshot was intended to provide more information about unit status than could be gained by merely freezing a Plan View.

The objectives of this portion of the effort were to:

1. test and refine utilities that allow users to pan and zoom map displays;

2. test and refine utilities that allow users to move in time through an exercise;

3. decide if there is a need for additional types of map displays; and

4 test and refine the integration of terrain data with network data.

Display of Control Measures

UPAS includes a utility that helps users to load data on unit control measures such as assembly areas, phase lines, and check points. The names and locations of these measures can be taken from the unit's graphics. Figure 7 shows a map display including unit control measures.

The initial interface for loading information on control measures required users to type the coordinates for each control measure. Figure 8 shows the amount of time required to load the names and coordinates of point control measures into the UPAS. This figure underestimates the time required to input data on control measures, because loading data on control measures in the form of points is less time consuming than loading data on control measures in the form of lines or areas. We concluded this process was too slow and prone to errors and decided that a mouse might be used as an input device for drawing the locations of control measures, removing the need to type in coordinates. The capability to use a mouse interface in loading control measures was subsequently implemented in an effort funded by STRICOM.

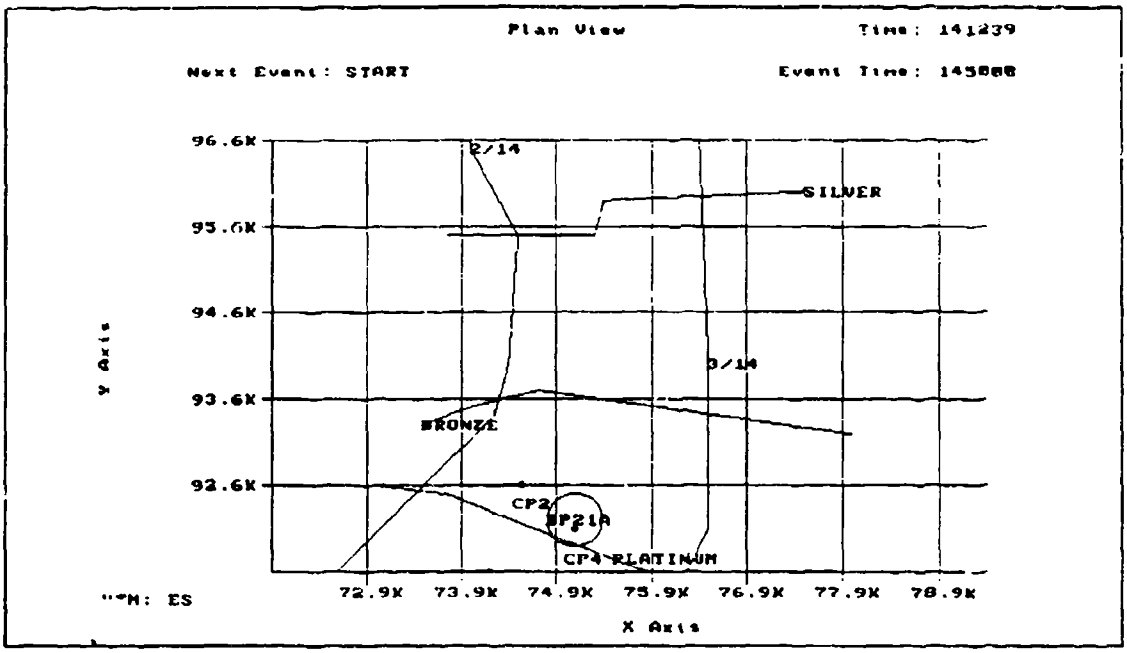


Figure 7. UPAS Map Display with Unit Control Measures.

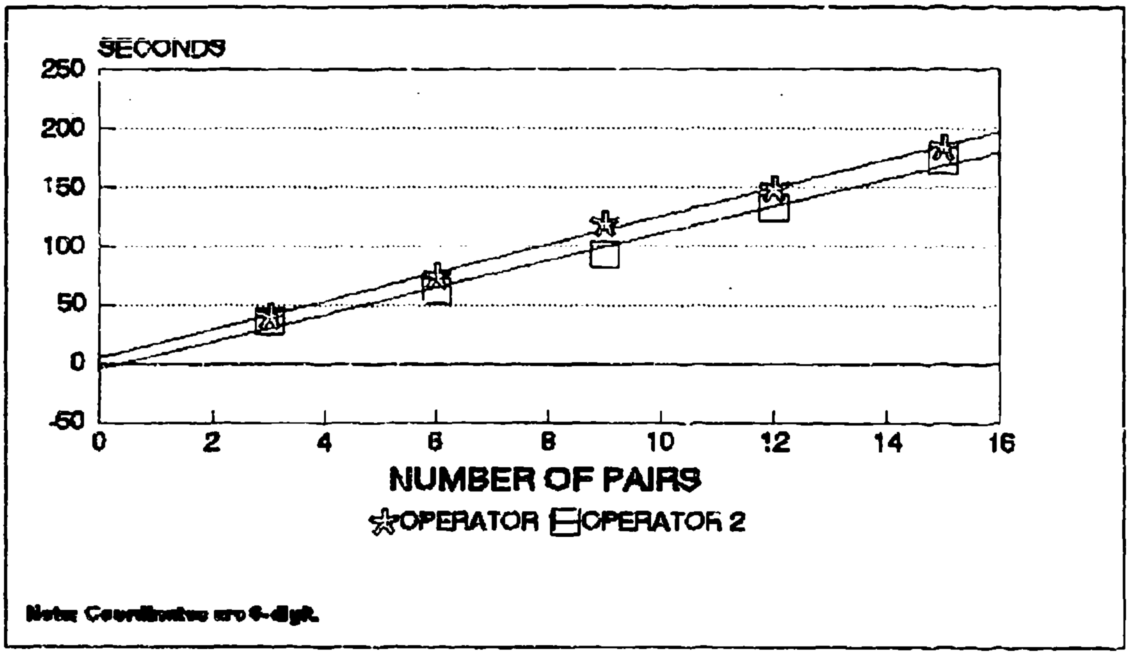
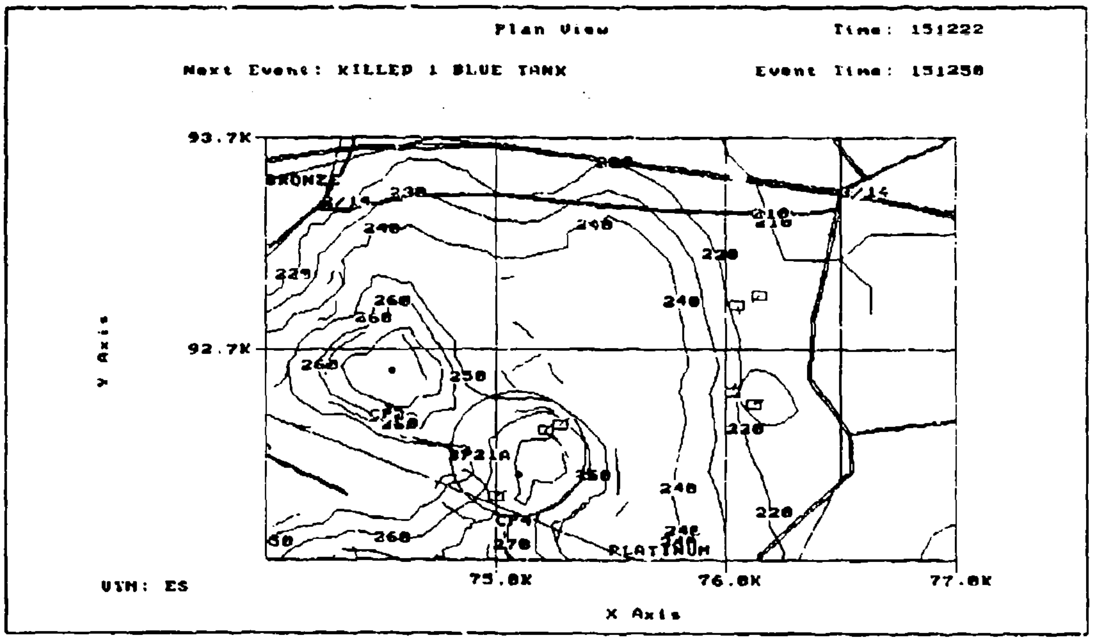


Figure 8. Time required to load data on unit control measures by typing coordinates of points.

Display of Terrain Data

Our initial attempt to add terrain data to map displays was limited to major terrain features (highways, unimproved roads, buildings, bodies of water, treelines, and clumps of trees), other than contour lines. This initial limitation was necessary because of the 640K DOS limit. We found that terrain data without contour lines was of limited value in terms of explaining the effects of terrain on unit performance. By reprogramming the UPAS to use less RAM, we were able to add the capability to display contour lines.

The issue of the most appropriate contour interval arose. SIMNET paper maps use a ten meter interval for contour lines. We decided that larger or smaller intervals might be required for certain applications. To support such applications, we decided to allow the user to type the desired interval or use the default of ten meters. We also decided that some users might want the elevation data to be displayed, while others might not want numbers to be included in the display of contour lines. The UPAS offers users the option of deciding whether to display elevation data for each contour line. Figure 9 shows a Plan View screen with contour lines. Contour lines are shown in the color magenta.



Plan View

Time: 151222

Next Event: KILLED 1 BLUE TANK

Event Time: 151250

UTM: ES

Figure 9. UPAS Map Display with Contour Lines.

The amount of time required to call up terrain features is a major variable influencing how map displays are used. The time required for the UPAS to display terrain features is a function of the size of the area, the concentration of terrain data within a particular area, and the operating speed of the computer. Figure 10 shows the time required to display terrain data as a function of the size of the area covered in a case where concentration of terrain features is a controlled variable. The procedures used to develop these data are presented in Appendix A. In cases where larger areas are covered, the display time can be quite long; therefore, the initial display of each map does not include terrain features. Because of the time required to call up terrain, the user should focus on that part of the battlefield of interest, before calling up terrain.

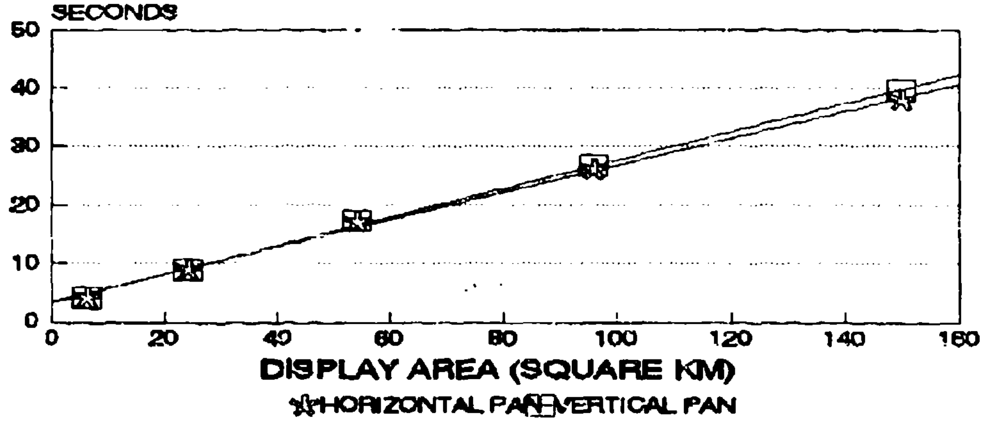


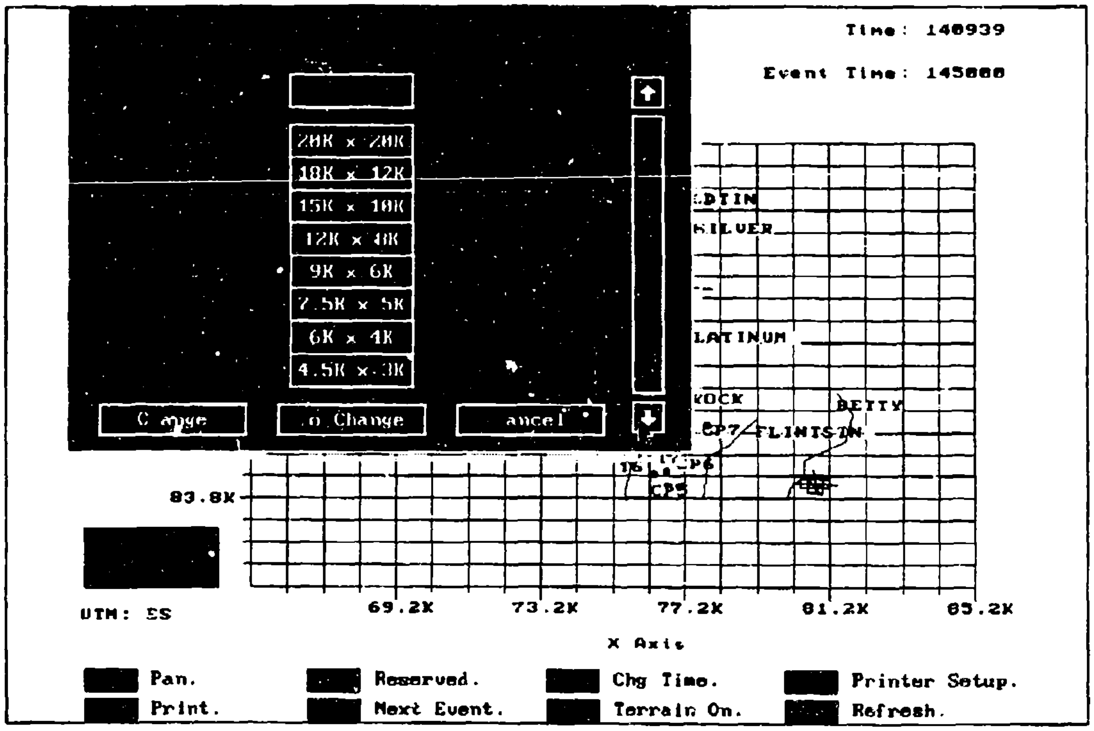
Figure 10. Time required to display maps as a function of the size of the area displayed.

Capability to Pan and Zoom

The capability to pan and zoom was a part of the initial Plan View, and it was included in new map displays. The interface for panning and zooming was awkward and slow. After selecting the change origin option, the map display disappeared and users were asked to type new coordinates for the X and Y axes. Users were then returned to the map display and had to select the "change scale" option. The map display would again disappear and users were prompted to type the new scale for the X and Y axes. After typing new scales, the map was redisplayed.

This interface became even slower and more cumbersome after terrain features were added to map displays, because a substantial amount of time was required to generate terrain features each time the map was redisplayed. The first change we made was to create an intermediate screen that allowed the user to make a variety of changes (e.g, change the origin, scale, and point in time of the replay) before the map is redisplayed. In addition, the display of terrain features was made an option so that users could focus on a particular part of the map before calling up these features.

Despite changes in the software, the tools for panning and zooming map displays were considered to be slow and awkward to use. ARI decided that integrating a mouse interface would be the best means of making it easy to pan and zoom. IST implemented a mouse interface. In the new system, the user moves an icon to a desired location in the center of a new display and presses a mouse key. A menu of scale options appears on screen (see Figure 11), including the option of typing in a scale not covered by the options.



Time: 140939

Event Time: 145000

Figure 11. Screen for selecting scales when panning and zooming with the mouse interface.

Application to Other SIMNET Terrain Databases

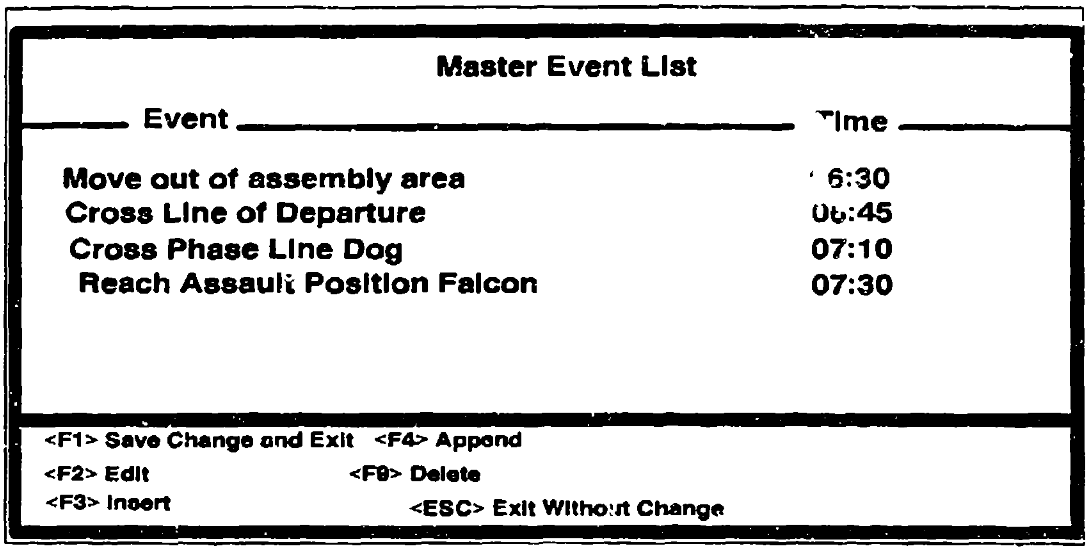
The SIMNET terrain database for the Fort Knox training area was used in the development of the UPAS. IST had previously developed PC-based software for converting SIMNET terrain data from the UNIX format to a DOS format as part of their work on the development of Intelligent Semi-Automated Forces. The DOS version of the Fort Knox database produced using this software was employed in the UPAS project. At present, two other SIMNET databases, Fort Hunter Liggett and Kuwait, have been transferred to a PC DOS format and are available for use with the UPAS. Additional databases might be converted at a low cost using the government owned software.

We had originally assumed that the UPAS could automatically employ any SIMNET terrain database translated to the appropriate PC DOS format. However, we failed to consider the unusual format of SIMNET maps in which coordinates are defined using a mixture of letters and numbers. The original UPAS software for displaying grids was designed to fit the letter combinations at Fort Knox. IST modified the UPAS software to allow it to display grids for any combination of letters and numbers. As a result, the UPAS can now use any of the SIMNET terrain databases that have been translated to the appropriate PC format.

Changing Exercise Time

The Plan View offers two ways to change exercise time during replays. First, users can move forward or backward to a specific time by typing in a new time. Second, the UPAS Plan View can move forward to points in time addressed by a Master Event List (MEL). The MEL allows users to record time-tagged events (see Figure 12). These events may be from the unit operations order (e.g., the time a unit is expected to cross its Line of Departure), or they may be based upon observations made during an exercise. The MEL is automatically tied to the Plan View, allowing the user to move from one time-tagged event to the next.

We found that changing time was a slow process with the original Plan View. Whichever method of changing time was employed, the map display disappeared from the screen while the UPAS searched for the first PDU with a time that matched the new time. The system had to examine time data within each PDU in sequence until it reached the first PDU having the desired time. The amount of time required to find the appropriate PDU was a function of the number of PDUs and the operating speed of the computer. In many cases moving just ten minutes forward in an exercise required ten minutes of execution time.



Master Event List

Event	Time
Move out of assembly area	06:30
Cross Line of Departure	06:45
Cross Phase Line Dog	07:10
Reach Assault Position Falcon	07:30

<F1> Save Change and Exit <F4> Append
<F2> Edit <F9> Delete
<F3> Insert <ESC> Exit Without Change

Figure 12. Master Event List (MEL)

The problem with the amount of time required to move from one point in the exercise to another was addressed by modifying the UPAS software to create a time index file for each exercise. This file identified the number of the first PDU occurring at the beginning of each one minute time period. The Plan View was then reprogrammed to use the time index when hunting for PDUs with the correct time. As a result of these changes, the user could move from any point to another in a replay within a few seconds.

All of the UPAS map displays were programmed initially to work with one minute time increments. We found that in many cases we needed resolution down to the one second level to apply measures of unit performance, and we changed all of the map displays to work with one second increments. The Plan View could be used to move from one time to another down to a specific second, Snapshots of the exercise could be created for a specific second, and Battle Flows could be used to mark the position of vehicles at intervals as small as one second. Implementing one second resolution in the displays increased the time required to move from one time to another, because it forced the program to read the time within each PDU between the one minute markers to find the desired second. To address this problem, the UPAS time index function was revised to index PDUs at one second, rather than one minute, intervals. As a result, the time index function was reprogrammed so that the file would contain the number of the first PDU at the beginning of each one second period.

Given our concern with preparing AAR aids as rapidly as possible after an exercise, it is important to note that the difference in time required to generate a one minute versus one second index file was trivial. Table 6 shows times required to create a one second time index file for exercise files of differing sizes as a function of the computer employed. Unlike the finding with data conversion time, a 486 computer speeds up the indexing process. For each case in which a 486 computer was employed the time required to prepare the index was reduced.

Table 6.

Time Required to Generate a Second by Second Index File of PDUS as a Function of Exercise File Size and Computer.

File Size (megabytes)	386-40Mz Index Time (minutes:seconds)	486-50Mz Index Time (minutes:seconds)
4.1	00:49	
8.8	00:44	00:33
15.4	1:19	
20.9	1:49	1:17
25.4	2:13	
34.4	3:14	
35.4	3:03	
39.0	3:24	
44.7	3:57	2:50
52.5	4:37	
54.4	5:09	
63.2	5:32	
68.6	6:31	
80.9	7:06	
86.6	7:44	
89.2	7:51	

Switching Between Map Displays

The major differences among the map displays are in terms of the type of network data displayed and the manner in which these data are displayed over the map. Due to the time required to focus on a specific part of a map, it would be efficient to change the type of map display (i.e., change the display of network data) without leaving the map. Unfortunately, moving from one type of map display to another in the UPAS requires exiting one display completely before entering another type.

The Need for Additional Types of Map Displays

Shlechter and Bessemer (in preparation) identified the need for a fourth type of map display to provide a summary of direct and indirect firing events. The need for this display capability was initially identified by one of the trainers interviewed at the Fort Knox CATTC. This display is intended to provide information about how unit fires were distributed as a function of terrain features and control measures over a period of time selectable by the user. More will be said about this display, the Firefight, later in the report.

Summary

Five problems common to the UPAS map displays were addressed. First, terrain features, including contour intervals, have been added to map displays. Second, we modified the UPAS to support any SIMNET terrain database that has been ported to the PC format used by the UPAS. Third, the capability to navigate in time through exercise replays has been facilitated by the design and implementation of time index files that allow the system to move quickly from one point in an exercise to another. Fourth, the potential applications of map displays to performance measurement have been increased by displaying unit control measures. Fifth, the time and effort required to pan and zoom the displays have been reduced by integrating a mouse interface with the UPAS.

In discussions with trainers, we discovered the need for a new type of map display of firing events which we called the "Firefight". We implemented the concept of a Firefight display in software, as described in the next section of this report.

One major problem in UPAS map displays remains to be addressed. This problem concerns the time and effort required to move from one type of map display to another. Once a user has focused a map on a particular part of the battlefield and selected a unit of interest, it would save substantial time and effort to provide users the capability to move to another type of map display without changing all of the settings.

Implementation and Refinement of UPAS Map Displays

Testing and refinement of each type of map display was conducted to meet two objectives. The first was to insure the display was user-friendly, and the second was to improve the capability of the display to support unit performance measurement. Refinements in displays were based largely upon user comments (Shlechter and Bessemer, in preparation).

Plan View

Tactical vehicles are represented by rectangular icons for which weapon system orientation is indicated by a line representing the gun tube. Friendly blue force (BLUFOR) vehicles are represented by blue rectangles, and enemy red force (REDFOR) vehicles are represented by red rectangles. Each time a vehicle fires, it brightens in color briefly. Vehicles that become casualties change color permanently (blue to cyan and red to white).

Identification of Units and Vehicles. The lack of a system for identifying units and vehicles in the Plan View proved to be a problem. Without a method of unit identification the user is forced to use the Battle Snapshots and Battle Flows to try and decide which unit is which when observing the Plan View. Due to the high update rate for vehicle appearance data it is not feasible to impose vehicle legends on the Plan View screen. Therefore, we decided to allow the user to use a mouse interface to select vehicles for which unit and vehicle ID information is required. This solution is currently being implemented.

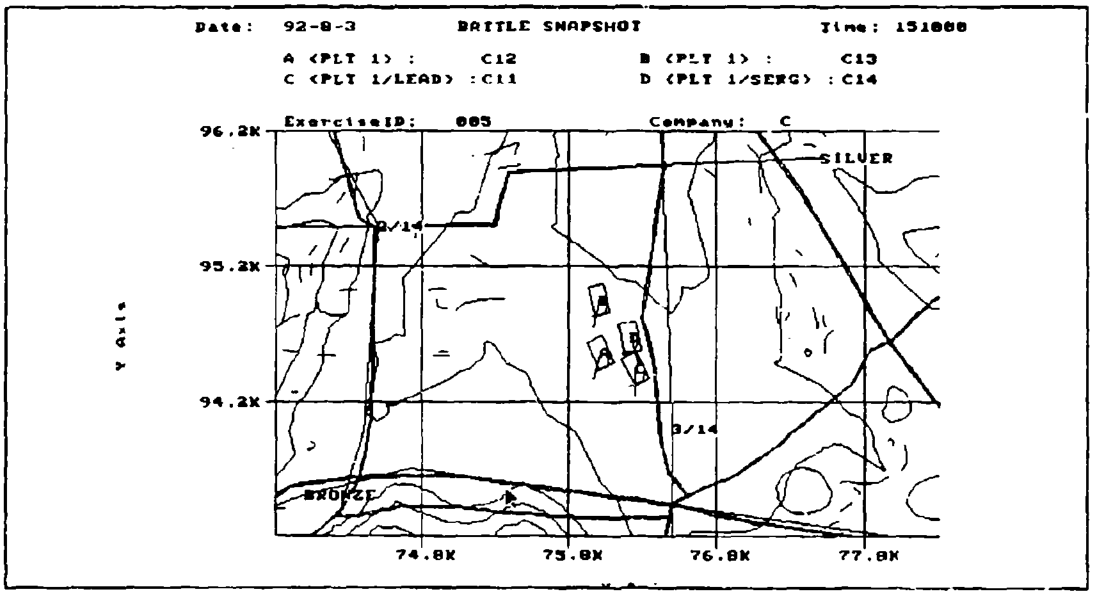
Controlling the Speed of Replays. We encountered problems with the speed with which the Plan View replays the exercise for individuals who want to use the Plan View to scan through an exercise looking for a critical event. The speed at which the Plan View replays an exercise depends on the operating speed of the computer and the rate at which data packets must be read. At points in the exercise when few packets are produced, the Plan View may replay the action at speeds greater than real time. At points when many packets are being generated (e.g., when many vehicles are moving quickly and firing), the replay slows down. We decided to add a fast forward capability to the Plan that works by reading every fifth packet, and this solution is currently being implemented in the UPAS.

Deciding What Targets Vehicles are Firing At. One of the refinements requested by users was the addition of shot lines. The addition of this information to the Plan View will make it possible for users to find out where shots are being directed. IST is currently revising the UPAS Plan View to meet this user requirement.

Battle Snapshot

Once again, the Battle Snapshot is capable of providing information about unit status that would be difficult to present in a display, like the Plan View, that is being updated constantly. The Snapshot was intended to provide an accurate representation of vehicle and gun tube orientation at a specific moment. In addition, it was expected to provide information about line-of-sight between or among vehicles.

Identification of Individual Vehicles and Units. The initial version of the Battle Snapshot allowed the user to specify the specific BLUFOR company or platoon to be observed. Each vehicle was marked with a letter, and a legend at the top of the screen was used to identify the bumper numbers associated with each letter. The Snapshot automatically displayed the location of all REDFOR vehicles, without identifying REDFOR vehicles by unit or bumper number. Further, icons for REDFOR vehicles were much smaller than those for BLUFOR. The difference in the display of BLUFOR and REDFOR data allowed the displays to focus on BLUFOR units. See Figure 12 for an example of a Battle Snapshot.



Date: 92-8-3 BATTLE SNAPSHOT Time: 151000

A <PLT 1> : C12 B <PLT 1> : C13

C <PLT 1/LEAD> :C11 D <PLT 1/SERG> : C14

ExerciseID: 005 Company: C

Figure 13. Example of a Battle Snapshot screen.

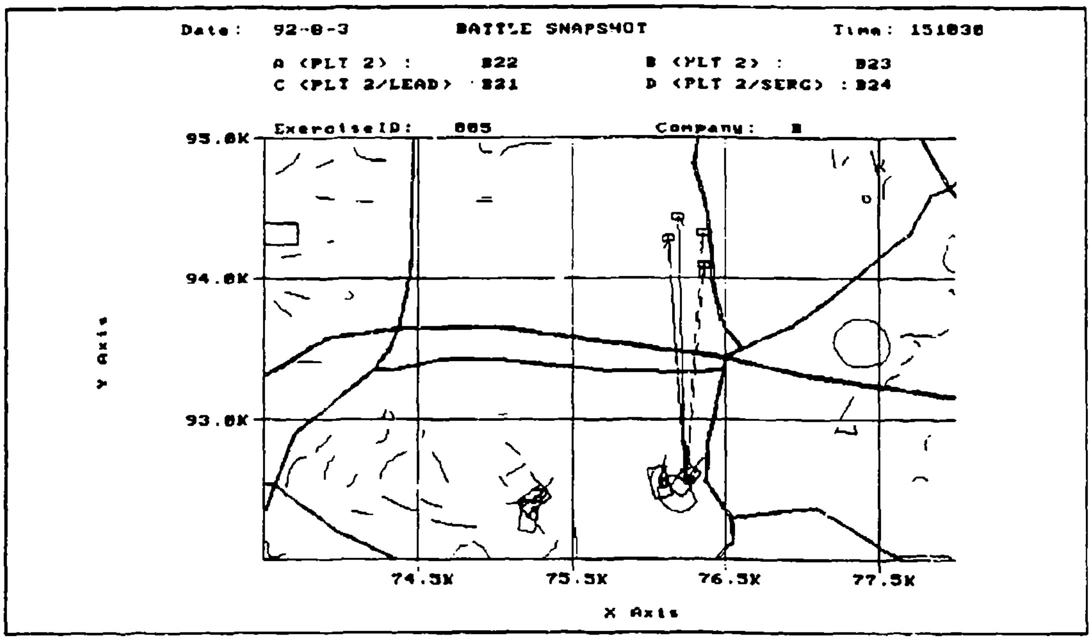
The first refinements in the Snapshot involved expanding the display options to company level and adding the capability to focus displays on REDFOR units. In moving to company-level displays, we were concerned that the screen would be cluttered with vehicle IDs. Therefore, we chose to display the vehicles for the Company Commander, XO, Platoon Leaders, and Platoon Sergeants only.

One problem remains to be addressed regarding the identification of individual vehicles. Trainers have expressed a preference for a different labeling system in which vehicles within a platoon are identified by one digit codes. The Platoon Leader would be "1," the leader's wingman "2," the Platoon Sergeant "4," and the Platoon Sergeant's wingman "3."

Time Intervals. The initial UPAS allowed the user to specify, to the nearest minute, when a snapshot was to be created for a specific unit. We discovered that many important events take place in less than a minute. Therefore, the Snapshot was revised to allow the user to specify time to the nearest second.

Line-of-Sight. The methods by which the UPAS calculates line-of-sight are described in Appendix C and in a related report by Petty and Campbell (1992). In the line-of-sight (LOS) display, LOS is indicated by a solid green line and a broken red line indicates that LOS is blocked by terrain. In the original version, the color coding was not used to indicate LOS in an all-or-none fashion, and a line might be red part way and green part way depending upon where LOS was obstructed. For example, if LOS is broken by a terrain feature immediately next to the source vehicle, then the entire line would be red. If LOS was broken by a terrain feature midway between the source and target vehicle, the line would be green until it reached the obstruction and then turn to red. The flaw in this coding method is its inability to address effectively situations in which the LOS is blocked by a terrain feature in the immediate vicinity of a target vehicle. In such cases, the entire line appears green to the naked eye, but a tiny portion of the line is, in fact, red. To address this problem, we reprogrammed the Battle Snapshot to display LOS in an all-or-none fashion. An example of a Battle Snapshot showing LOS and lack of LOS is shown in Figure 14.

The LOS display implemented within the UPAS could be used for examining LOS between friendly and enemy vehicles only. We considered this to be a serious deficiency because there are times when a trainer or researcher wants to know if LOS exists between two friendly forces. For example, when one unit is covering the movement of another unit is important to know if the covering force can see the force they are intended to protect. This deficiency in measurement capability is being addressed by changing the software to allow LOS to be displayed between friendly vehicles.



```
Date:   92-8-3          BATTLE SNAPSHOT               Time: 151038

        A (PLT 2) :          B22        B (PLT 2) :          B23
        C (PLT 2/LEAD)       B21        D (PLT 2/SERG) :     B24

        ExerciseID:     005             Company:   B
```

Figure 14. Sample Battle Snapshot screen with Line-of-Sight (LOS) Display.

In cases where a single vehicle is to be designated as the target vehicle, UPAS requires the user to move a rectangular icon around the screen with arrow keys until it covers the area in which the desired vehicle is located. UPAS will provide a legend at the bottom of the screen reporting the number of vehicles covered by the icon. The user must then repeatedly press the space bar to cycle through a list of the IDs of the vehicles covered by the icon and press <Enter> when the desired vehicle number is displayed. This is a slow and tedious procedure. We decided that the LOS feature could be made easier to use by allowing the user to employ a mouse device to select and verify the vehicle of interest, and this change was implemented.

In addition to problems in displaying LOS data, we encountered problems in deciding how to employ LOS in measuring unit performance. LOS between an enemy and friendly vehicle is both good and bad. Our solution involves using patterns of LOS that might provide a meaningful assessment of unit performance. One such measure is to consider how many vehicles from a platoon have LOS when contact is made with the enemy.

Battle Flow

The UPAS Battle Flow is an animated figure that traces the movement of vehicles and units throughout the course of a mission or during a significant segment of a mission. The trace is implemented by showing vehicle locations at a predetermined interval, such as every minute. The icons used to represent vehicle locations are plus signs and numerals. The user can stop at any point during the trace of the movement of vehicles and create a hard copy of the display.

The Battle Flow allows the user to specify the time interval at which vehicle positions are to be marked. This important feature allows one to adjust the position updates to avoid cluttering the screen with data. The Battle Flow also allows the user to select the interval over which a trace is to be prepared. Like the Snapshot, the Battle Flow allows the user to specify the unit to be examined in the trace. See Figure 15 for an example of a Battle Flow.

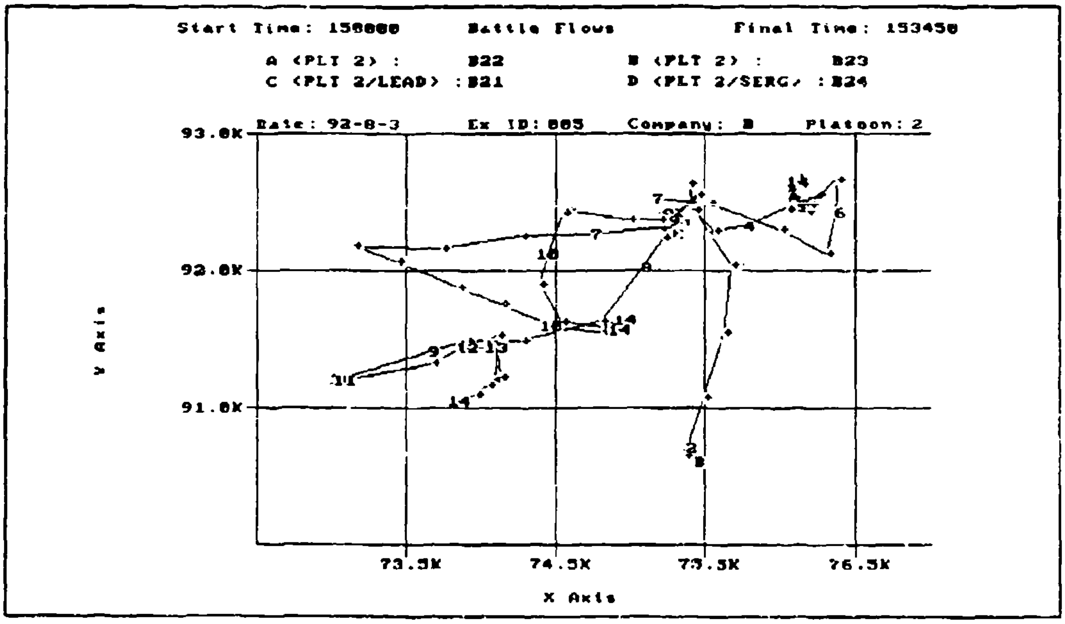


Figure 15. Sample Battle Flow Display.

We envisioned the Battle Flow as a tool for examining the performance of units in offensive roles, but we found that it can also be useful when examining the performance of units in defensive roles. The source of this discovery was an exercise in which a unit repeatedly encountered problems in moving to the position it was intended to occupy. This observation led us to consider other cases where inadequate planning, reconnaissance, or dissemination of information in the context of a defensive mission might lead to wasted movements that might be illustrated with a trace. These cases include withdrawal to a supplementary position, occupation of alternate firing positions, withdrawal of Outposts (OPs) to the main body of their unit, and the return of scouts to the main body after their reconnaissance.

The initial Battle Flow allowed the user to specify the beginning time for a trace but not the end time, and the smallest time interval that could be addressed by a marker was one minute. These features limited the ability of the Battle Flow to trace movements occurring over short periods. The first exercise that raised our concern about addressing small time periods was one in which chaotic movement was observed during an assault. Rather than moving abreast of one another, vehicles continually crossed in front of each other. A trace of this unit's movement during the assault was considered to be an ideal AAR aid, but one minute time resolution did not allow us to provide the needed display. The Battle Flow was modified to allow the user to specify the end of the trace and employ position marker intervals as small as one second.

Fire Fight

A Fire Fight Display (Figure 16) shows direct and indirect firing events over a terrain map. The period in time covered by this display is user selectable. Direct firing events are displayed with shot lines connecting the location of the firing vehicle with the location of the vehicle or ground impact, and a vehicle icon is used to show the location of the firing vehicle using the same color coding system as the Plan View. A miss is indicated by a white line, and a green line indicates a hit or kill. If the firing event results in a kill, there will also be a dead vehicle icon at the target location (cyan for a destroyed BLUFOR vehicle and white for a destroyed REDFOR vehicle. Artillery impacts are shown using white rectangles.

The original version of the Fire Fight treated all types of rounds the same way. As a result, shotlines for 25 mm firing events tended to dominate the displays and obscured tank firing events. We decided to address this problem by using a sampling plan to reduce the shotlines for 25 mm rounds.

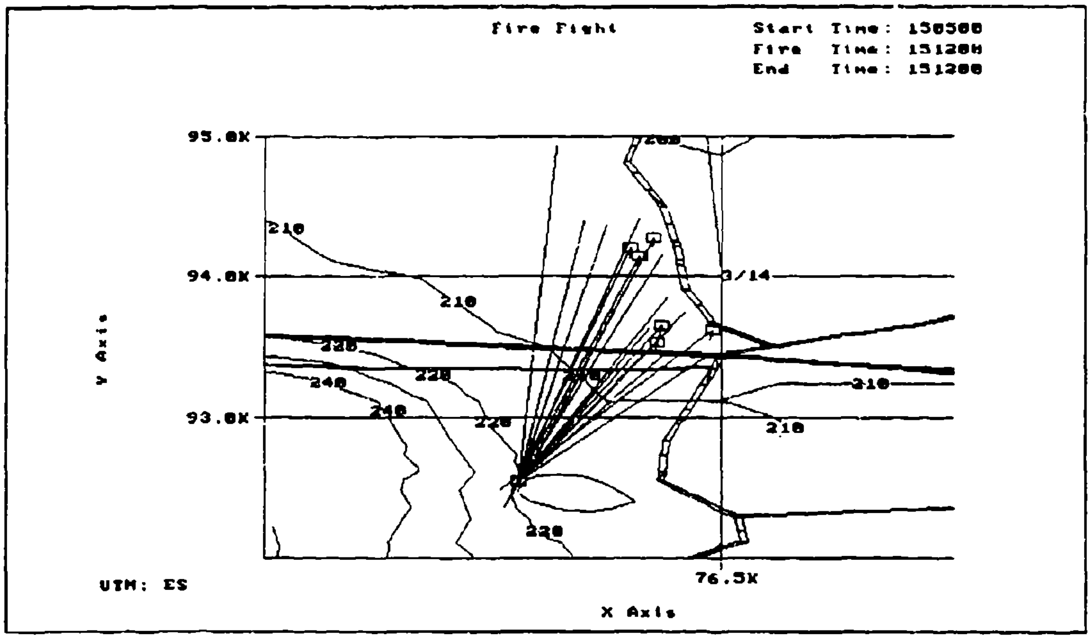


Figure 16. Sample Fire Fight Display.

Identifying the force associated with artillery impacts was a second problem encountered with the Fire Fight. Identification of the force to which a weapon system belongs is accomplished by taking information from its Vehicle Appearance PDUs. However, indirect fire missions can be created in the absence of Vehicle Appearance PDUs. For firing events in which the firer's logical player number is 0.0.0, no vehicle exists.

Summary

The utility of the Plan View Display to exercises above platoon level is limited by the lack of a capability to match vehicles on screen with specific unit and vehicle Ids. We are addressing this deficiency by adding the capability to call up vehicle and unit IDs by using a mouse to select vehicles on screen.

The utility of the Battle Snapshot and Battle Flow were increased by modifying these displays to address periods of time as small as one second. The utility of the Battle Snapshot and Battle Flow would be enhanced further if vehicles at platoon level were identified as 1 (Platoon Leader), 2 (Platoon Leader's wingman), 3 (Platoon Sergeant's wingman), and 4 (Platoon Sergeant). To accomplish this change, platoons, rather than individual vehicles, would need to be color coded.

After Action Review Presentation Manager

Problem and Short-Term Solution

The UPAS provides a variety of displays that can be used to illustrate key teaching points during AARs, but it does not provide tools for integrating displays to provide a polished presentation. If a trainer attempted to host an AAR using the UPAS, then there would be considerable down time as the trainer moved from one type of display to another, identified the unit of interest, moved to the point in time of interest, and used pan and zoom capabilities to focus on the activity of interest.

We saw the need for an "AAR Presentation Manager" that could be used to control the presentation of AAR aids in a way that supports the smooth flow of an AAR. The Presentation Manager should allow the operator to display screens (e.g., a Snapshot) created prior to the start of the AAR. The Manager should also allow the operator to move quickly from one data display to another.

Our first attempt to develop an AAR Presentation Manager involved using commercial off the shelf software to capture and sequence UPAS screens. One drawback with this approach is that it addresses static displays only. A second drawback is that the tools for capturing and sequencing displays are not integrated with the UPAS or the AAR process. For example, there was no method for providing saved screens with a descriptive label, and the user was forced to exit UPAS to view or use the screens.

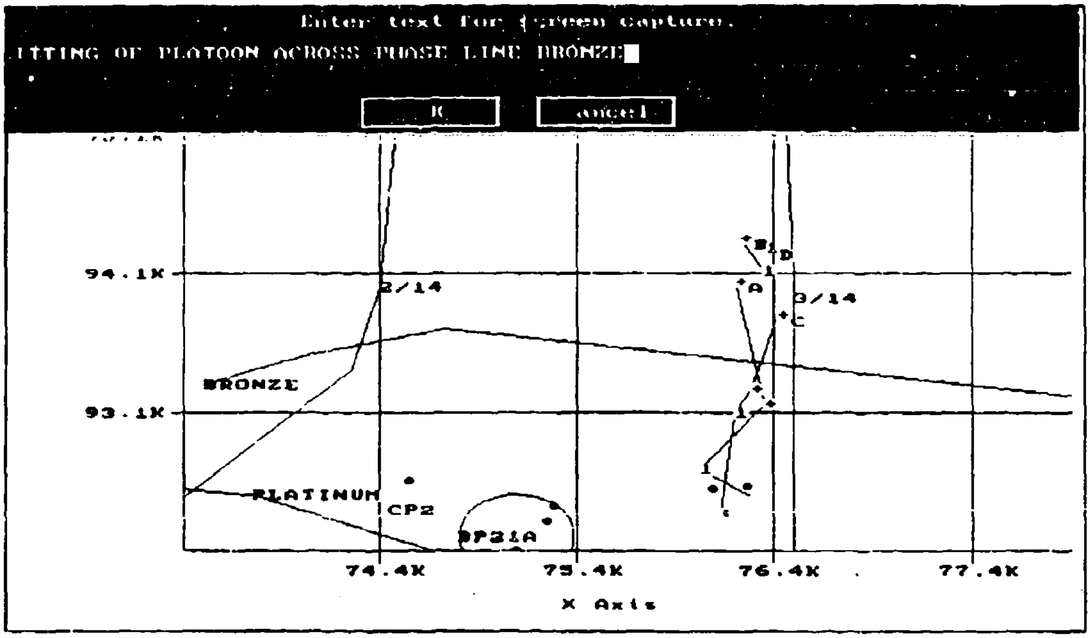
The type of UPAS presentation manager preferred by trainers is one that focuses on the replay of the exercise using the Plan View Display and allows the trainer to call up static displays (table, graph, Exercise Timeline, Battle Flow, Snapshot, Fire Fight) at appropriate points in the replay. In preparing for AARs, the trainer would create the status displays using UPAS utilities, save displays, and sequence displays.

The desired AAR presentation sounds like a typical "Windows"* application. However, there are problems implementing these applications in the UPAS. First, UPAS itself was not created to run under Windows. Second, the XDB database management system within UPAS is not compatible with Windows. We attempted running two DOS sessions under Windows. One session is devoted to running the UPAS Plan View, while the second shows data display screens captured using the WordPerfect "grab" utility. Under this approach, the user can observe the Plan View, move to the other session to view a static display, and move back to the Plan View to continue watching the replay. This approach is awkward and time-consuming to employ.

* Windows is a registered trademark of the Microsoft Corporation.

Intermediate-Term Solution

We revised the UPAS to fit the concept of an AAR Presentation Manager. A utility was added to the Plan View, Snapshot, Battle Flow, Firefight, and Timeline that allows the user to capture screens of interest in the "pcx" file format and type up to two lines of comments. Figure 16 was created using the "save screen" portion of the new AAR resentation Manager.



Enter text for screen capture.

ITTING OF PLOTOON ACROSS PHASE LINE BRONZE

Figure 17. Saved Battle Flow screen with teaching point from AAR Presentation Manager.

Saved screens are named according to the type of display (e.g., "BF0001" for the first Battle Flow screen) to help users identify file contents. The user gains access to saved screens through a menu of these screens, as illustrated in Figure 18.

Although we expect the AAR Presentation Manager to address many of the problems in preparing polished, integrated displays for AARs, we know that this new tool will still fall short of meeting the requiremert of supporting AARs in a timely manner. Because the UPAS runs on a personal computer, it can perform only one task at a time. During an exercise, the UPAS is devoted completely to the task of collecting network data. The job of preparing and integrating data displays for an AAR cannot begin until after an exercise is over.

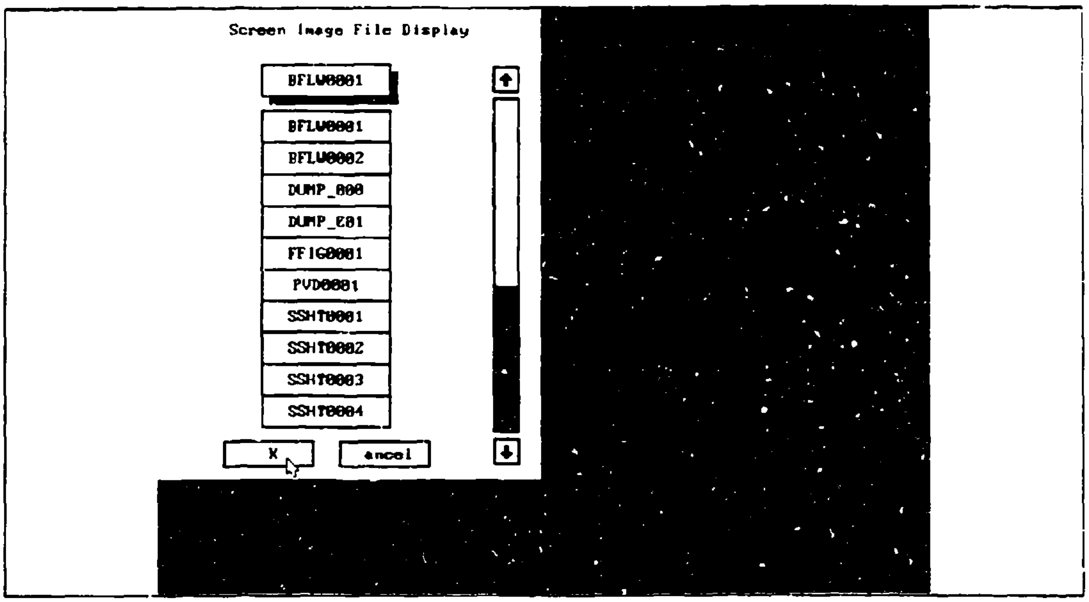


Figure 18. Screen Image File Display menu from the AAR Presentation Manager.

Summary

Integrating diverse types of data displays into a single medium to support AARs is an important goal. Regardless of the value of the information contained in individual data displays, these displays are unlikely to be used to support AARs unless we can provide the user with a means to prepare and examine candidate data displays, select displays, sequence displays, and integrate displays promptly after an exercise.

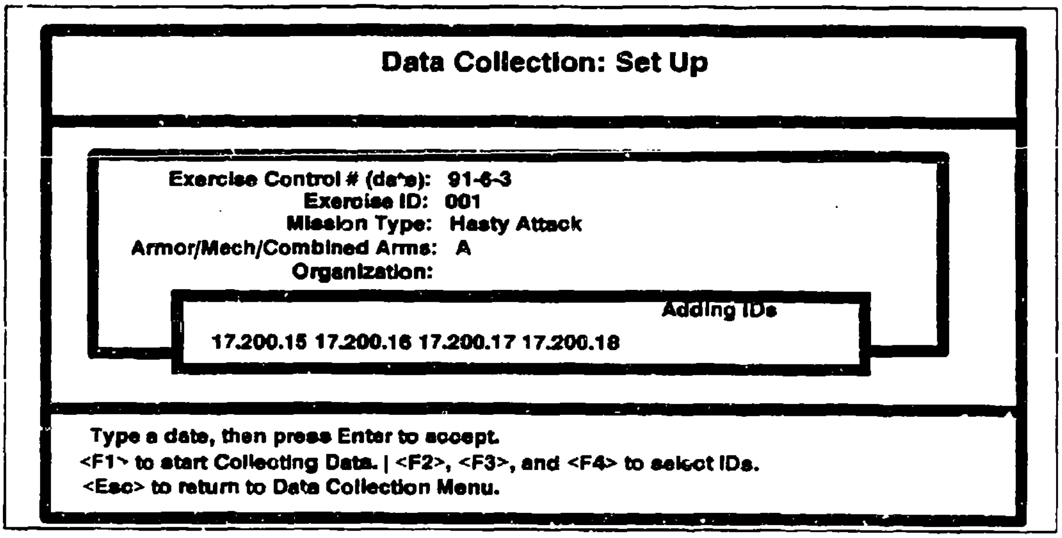
The utility of the AAR Presentation Manager currently being implemented in the UPAS is limited by the fact that the AAR preparation process cannot begin until after an exercise is over, because the UPAS is fully occupied collecting network data during exercises. Future porting of the UPAS to a work station environment would allow the process of preparing data displays for AARs to begin during an exercise.

Time Required for Operator to Load Non-Network Data

Certain information required to guide data collection and integrate planning data must be loaded by an operator. These four major data loading activities are described below.

Limiting Network Data Collection to Specified Vehicles

In cases where multiple exercises are conducted concurrently under the same Exercise ID, the operator may want to limit network data collection to specific vehicles. This is accomplished by typing the logical player numbers of vehicles for which data are to be collected. These numbers have three parts (the side ID, the host ID, and the vehicle number) separated by a period, such as "17.200.15." Figure 19 shows the UPAS screen used to load this information, and Figure 20 shows the amount of time required to load these data as a function of the number of vehicles for two sample operators.



Data Collection: Set Up

Exercise Control # (date): 91-6-3
Exercise ID: 001
Mission Type: Hasty Attack
Armor/Mech/Combined Arms: A
Organization:

Adding IDs

17.200.15 17.200.16 17.200.17 17.200.18

Type a date, then press Enter to accept.
<F1> to start Collecting Data. | <F2>, <F3>, and <F4> to select IDs.
<Esc> to return to Data Collection Menu.

Figure 19. Data collection set up screen for limiting data collection to specific vehicles.

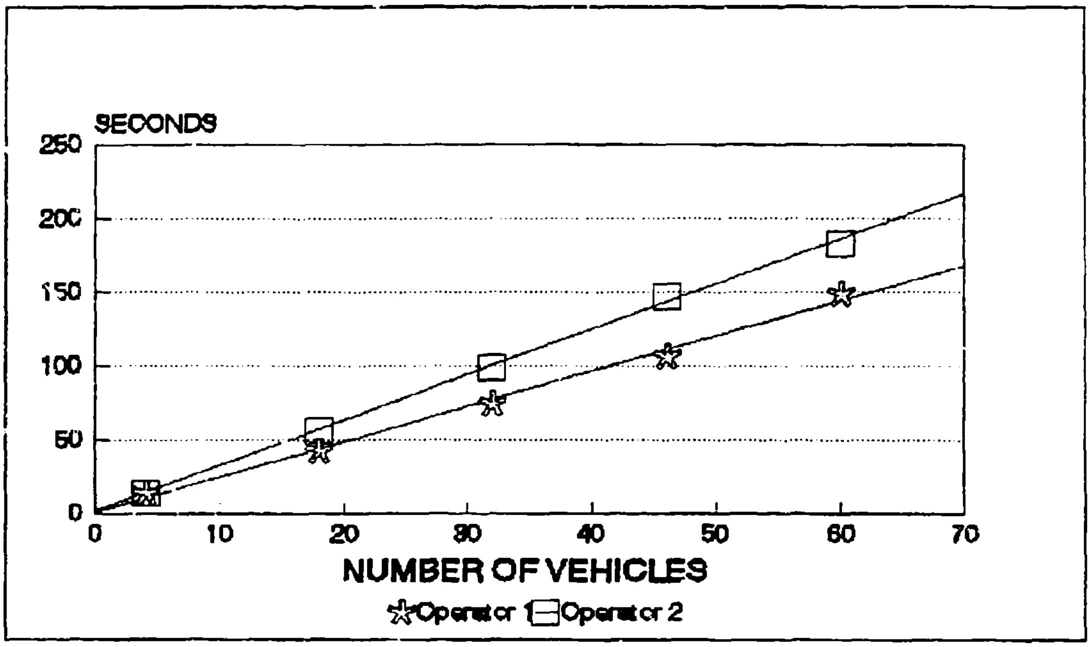


Figure 20. Time required to specify vehicles for which data are to be collected.

Master Event List

The Master Event List utility allows the user to type in time-tagged exercise events. These events may be from the unit's operations order, critical events observed during an exercise, or a combination of types of events. Loading this information is useful because the MEL can be used to guide the replay of the exercise automatically using the UPAS Plan View display. The amount of time required to load information into the MEL as a function of the number of characters loaded is shown for two sample operators in Figure 21.

Unit Control Measures

Loading the names and locations of unit control measures into the UPAS makes it possible for the system to display these control measures in all of the map displays, and it makes it possible to use the Exercise Timeline to find out when a unit crosses each control measure. These control measures include points (e.g., check points), lines (e.g. phase lines), and areas (e.g., objectives). Figures 22 and 23 show the times required to load information on lines and areas, respectively.

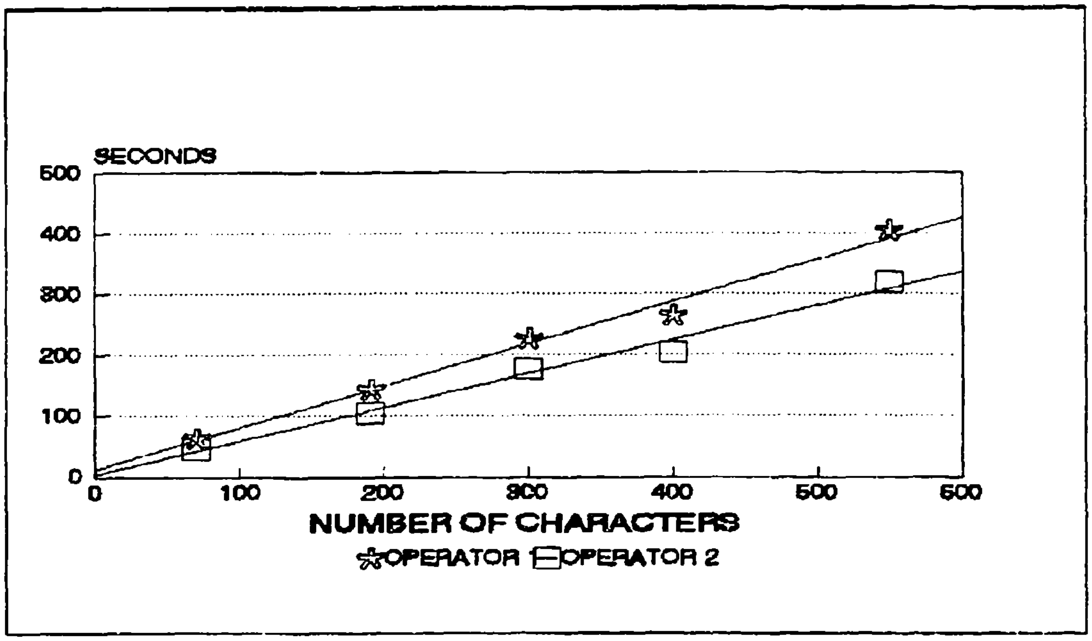


Figure 21. Time required to load Master Event List with information on key time-tagged exercise events.

We expect that the recently implemented mouse interface for loading control measures reduces the time required to load these data. Regardless of the speed issue, the mouse interface for loading control measures is much easier to use than the previous version.

Loading Information on Platoon Organization

The UPAS requires information about which vehicles are assigned to each platoon and which platoons fall within each company to allow the user to focus displays on a specific unit. The UPAS also requires information about which vehicles are manned by company commanders, XOs, platoon leaders, and platoon sergeants. The user must enter this information using the screen illustrated in Figure 24. To reach such a screen the user must go through a series of menus to identify the side and company for which a platoon is being defined.

The amount of time required to load information on platoon organization is shown in Figure 25. The time required to load these data is longer than we would like, but we have not developed an idea for a more rapid interface for performing this function.

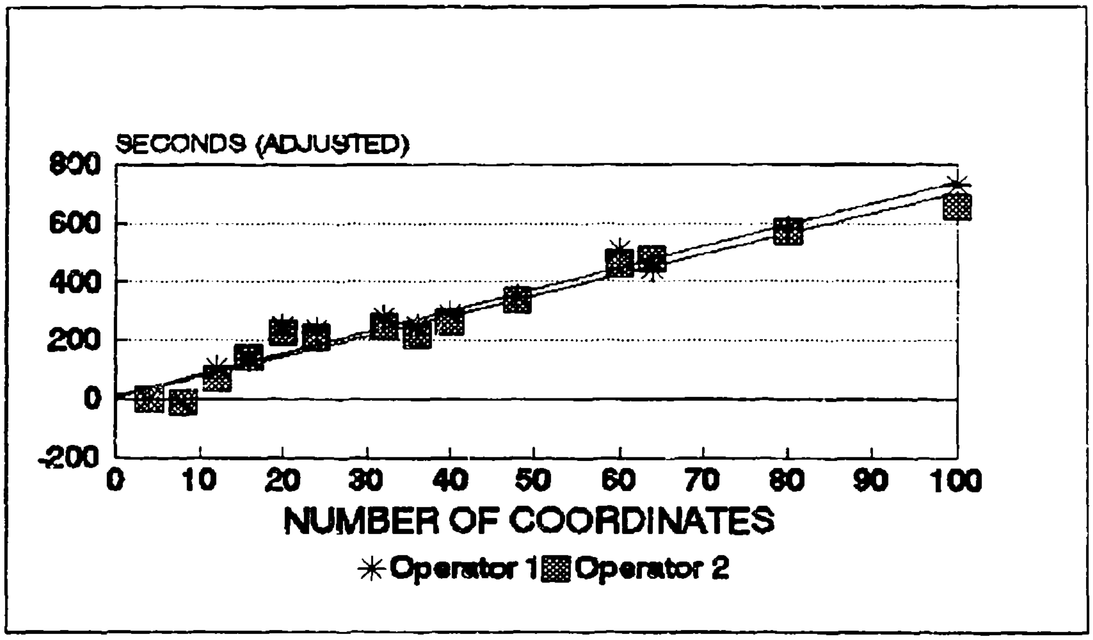


Figure 22. Time required to load information on control measures that are lines.

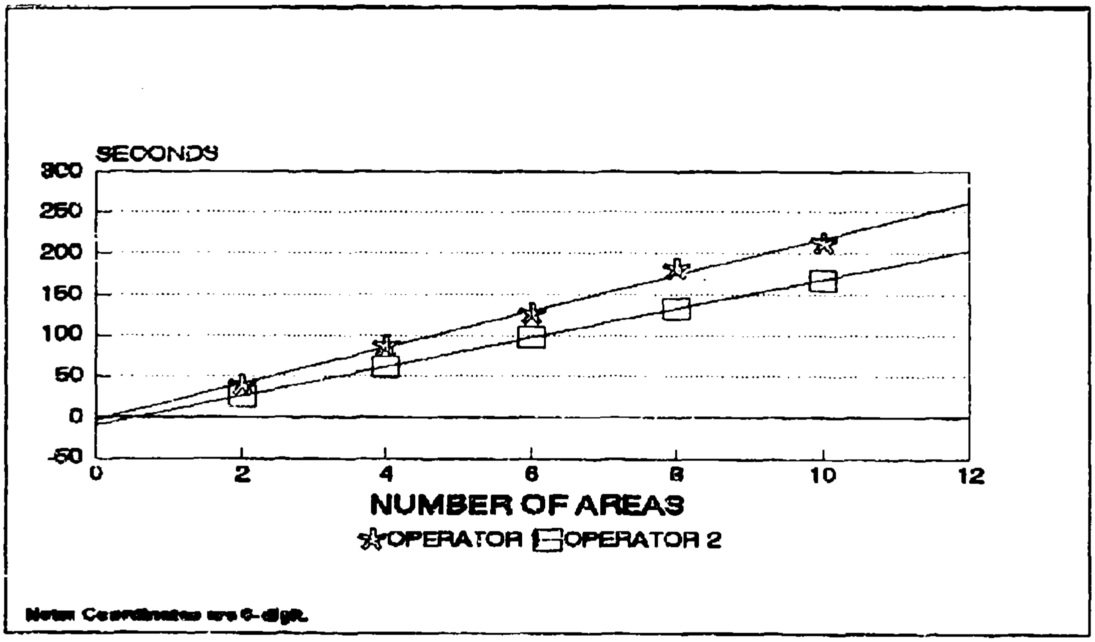
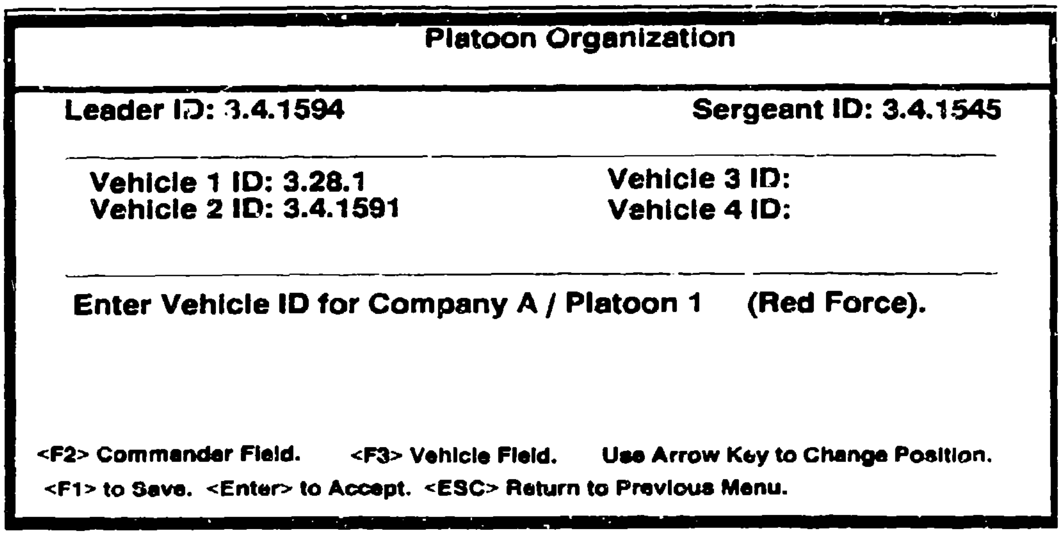


Figure 23. Time required to load information on control measures that are areas.



Platoon Organization

Leader ID: 3.4.1594 Sergeant ID: 3.4.1545

Vehicle 1 ID: 3.28.1
Vehicle 2 ID: 3.4.1591

Vehicle 3 ID:
Vehicle 4 ID:

Enter Vehicle ID for Company A / Platoon 1 (Red Force).

<F2> Commander Field. <F3> Vehicle Field. Use Arrow Key to Change Position.
<F1> to Save. <Enter> to Accept. <ESC> Return to Previous Menu.

Figure 24. Screen for loading data on platoon organization.

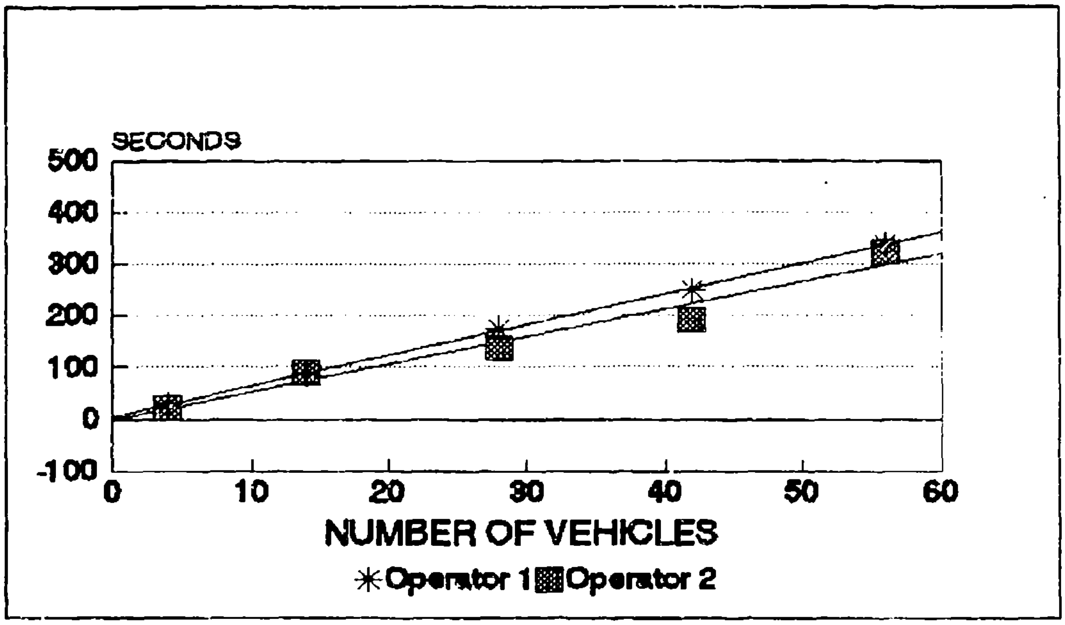


Figure 25. Time required to load information on platoon organization.

Integration of UPAS Lessons Learned with Future Applications of Networked Simulations

Joint Service Standards for Distributed Interactive Simulation Feedback Systems

Lessons learned from the UPAS project have been used in developing standards for feedback systems as part of the series of joint service Workshops on Standards for the Interoperability of Defense Simulations sponsored by the Army's Simulation, Training, and Instrumentation Command (STRICOM) and the Defense Modeling and Simulations Office (DMSO) and incorporated into supporting rationale document (Exercise Control and Feedback Requirements Working Group, 1992).

Application to a Close Combat Tactical Trainer

The Close Combat Tactical Trainer (CCTT) will be the Army's next generation of networked simulator for armor and mechanized infantry, and the contract for a CCTT has already been awarded. The UPAS project has been briefed to key members of the CCTT project team, including the STRICOM Project Manager for Combined Arms Tactical Trainer (PM-CATT). According to a Memorandum of Agreement between ARI and PM-CATT, ARI will consult on the development of a CCTT AAR system based upon lessons learned from the UPAS project. ARI is represented on the concurrent engineering working group for CCTT work stations that include the AAR system.

Training Technology Base

The UPAS project is tied to three efforts to enhance the training technology base. One effort involves integrating an expert system with UPAS displays to guide the preparation of AAR aids. The second involves applying UPAS data displays to close air support (CAS) training, and the third involves integrating the UPAS with behavioral modeling software. Each of these efforts is described below.

An Automated Training and Feedback System (ATAFS) is being developed by LB&M Associates to support training feedback in the SIMNET environment. The ATAFS project will involve porting UPAS data displays to a work station environment, implementing an expert system that will guide the preparation of AAR aids during exercises, and developing an improved AAR Presentation Manager. LB&M designed the ATAFS concept to address many of the shortfalls of the UPAS described in the current report (LB&M Associates, 1992).

The UPAS will be upgraded to DIS protocols and various UPAS displays will be enhanced in the four-service Multi-Service Distributed Training Tested (MDT2) Project. This project is sponsored by the Defense Modeling and Simulations Office (DMSO), and its near term goal is to set up a testbed to support CAS training.

The "Behavioral Modeling and Simulation Adjuncts for Distributed Interactive Simulation (DIS)" Project also employs the UPAS as a tool for measuring unit performance. This project involves modifying behavioral modeling software to produce SIMNET Version 6.61 PDUs as output. The software is being used to model the performance of armor platoons on a SUN workstation, and the UPAS is being used to measure the behavior of the platoons generated by the modeling software.

Application to Measuring the Behavior of Improved Semi-Automated Forces (SAFOR)

The Army is working to improve the quality of SAFOR through the development of a series of Modular SAFOR (MODSAF). The UPAS was used to measure the performance of modular SAFOR (MODSAF) Version C the week of 13 September 1993. This application was supported and funded by the PM-CATT, the TSM-CATT, and the DMSO. The goals of this application were to compare the behavior of MODSAF and manned units and to assess whether MODSAF behavior was controlled by the intended physical parameters (e.g., firing controlled by LOS and range).

SUMMARY

The current version of the UPAS (Meliza and Tan, in preparation) described in Figure 26, differs significantly from the prototype illustrated on page 4 of this report. New components are indicated with an asterisk.

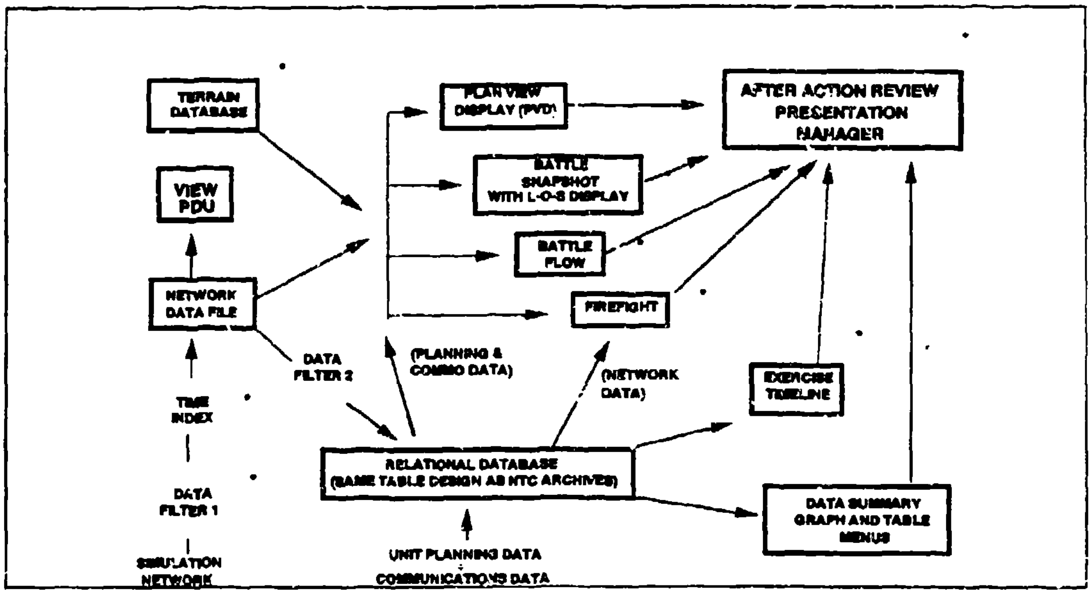


Figure 26. Overview of the enhanced UPAS.

Major additions to the UPAS are listed below.

1. The capability to limit network data collection to specific entities has been added through refinement of the UPAS data collection filter.

2. System capability to collect critical data during peak periods of network activity has been enhanced by increasing the rate at which data are loaded to disk and collecting critical PDUs at the expense of less critical PDUs.

3. The utility of the UPAS in navigating through exercise data has been increased by adding a time indexing file for PDUs and revising all map displays to use this index to move from one point in the exercise to another quickly.

4. Terrain data and unit control measures have been added to map displays to provide a more complete picture of unit performance than can be gained by merely replaying network data over a grid map.

5. Three new types of map displays have been added to the moment by moment replay to support unit performance assessment; a trace of movement over time, a detailed view of a unit's situation at a specific point in time, and a summary of firing events as a function of terrain features and control measures.

6. A new type of tool, the Exercise Timeline, has been created to examine unit movement, shooting, and communication events independently and in combination with each other.

7. An AAR Presentation Manager was implemented to help trainers use the UPAS data displays in an efficient manner.

In addition to adding new components to the UPAS, this effort has produced a number of refinements in existing UPAS capabilities. The most important of these changes is the capability to support unit performance assessment at company level as well as platoon level. Other important changes include the following:

1. modification of the UPAS to employ any SIMNET terrain database;

2. integration of a mouse interface to facilitate panning and zooming map displays;

3. integration of a mouse to allow users to identify vehicles and units when using the Plan View; and

4. implementation of an AAR Presentation Manager that allows trainers to save and call up static display screens (graphs, tables, snapshots, timelines, etc.) when replaying the exercise with the UPAS Plan View;

In the course of the UPAS project we have identified certain needs of trainers and researchers that could not be met, or have not been met, by refinements in the UPAS. These needs form part of the lessons learned that need to be addressed as future DIS AAR and performance measurement systems are developed. These lessons have been carefully noted in the appropriate sections of this report.

REFERENCES

Hiller, J. H. (1987). Deriving useful lessons from combat simulations. Defense Management Journal, Second and Third Quarter, 28-33.

Institute for Simulation and Training (1991). Military Standard: Protocol Data Units for Entity Formation and Entity Interaction in a Distributed Interactive Simulation. Orlando, FL: University of Central Florida Institute for Simulation and Training.

Institute for Simulation and Training (1993). Proposed IEEE Standard (Second Draft): Exercise Control and Feedback Requirements for Distributed Interactive Simulation. Orlando, FL: University of Central Florida Institute for Simulation and Training.

Kerins, J. W., Atwood, N. K., & Root, J. T. (1990). Concept for a common performance measurement system for unit training at the National Training Center (NTC) and with Simulation Networking (ARI Research Report 1574). Alexandria, VA: U.S. Army Research Institute for the Behavioral and Social Sciences. (AD A226 956).

LB&M Associates (1992). Phase I Small Business Innovative Research (SBIR) Program for Simulation Networking Training Performance Assessment Task 4 Report (Final Operational and Technological Concept). Lawton, OK.

Meliza, L. L., Bessemer, D. W., Burnside, B. L., & Shlechter, T.M. (1992). Platoon-level After Action Review aids in the SIMNET Unit Performance Assessment System (ARI Technical Report 956). Alexandria, VA: U.S. Army Research Institute for the Behavioral and Social Sciences. (AD A254 909)

Meliza, L. L., Tan, S. C., White, S., Gross, W., & McMeel, K. (1992). SIMNET Unit Performance Assessment System (UPAS) user's guide (ARI Research Product 92-02). Alexandria, VA: U.S. Army Research Institute for the Behavioral and Social Sciences. (AD A250 699)

Meliza, L. L., Tan, S. C., White, S., Gross, W., & McMeel, K. (in preparation). SIMNET Unit Performance Assessment System advanced user's guide. Alexandria, VA: U.S. Army Research Institute for the Behavioral and Social Sciences.

Meliza, L. L. (1993). SIMNET/Training Requirements Relational Database user's guide (ARI Research Product 94-01). Alexandria, VA: U.S. Army Research Institute for the Behavioral and Social Sciences. (AD A275 634)

Meliza, L. L., & Tan, S. C. (in preparation). SIMNET Unit Performance Assessment System (UPAS) version two user's guide. Alexandria, VA: U.S. Army Research Institute for the Behavioral and Social Sciences.

Petty, M., & Campbell, C. (1992). Efficient line of sight determination in polygonal terrain. IST Technical Report IST-TR-92-5. Orlando, FL: University of Central Florida Institute for Simulation and Training.

Pope, A. R., & Schaffer, R. L. (1991). The SIMNET network and protocols (BBN Report No. 7627). Cambridge, MA: BBN Systems and Technologies.

Shlechter, T. M., Meliza, L. L., Burnside, B. L., & Bessemer, D. W. (1992). Developing Automated Feedback Materials for a Training Simulator: An Interaction Between Users and Researchers. Paper presented at the 1992 Annual Meeting of the Association for Educational Research, San Francisco, CA.

Shlechter, T. M., & Bessemer, D. W. (in preparation). Evaluating and refining the Unit Performance Assessment System's After Action Review Displays. Alexandria, VA: U.S. Army Research Institute for the Behavioral and Social Sciences.

Thorpe, J. A. (1988). Warfighting with SIMNET--A Report from the Front. Proceedings of the 10th Interservice and Industry Training Systems Conference, pp. 263-273.

White, S., McMeel, K., & Gross, W. (1990). Unit performance measurement and training program design for networked simulators (PTR-4013-1-90). Woodland Hills, CA: Perceptronics.

XDB Systems, Inc. (1990a). XDB SQL reference. College Park, MD.

XDB Systems, Inc. (1990b). XDB user's manual. College Park, MD.

APPENDIX A

IMPROVED DATA COLLECTION AND FILTERING TECHNIQUES

It has been reported from field testing that, at times UPAS loses data packets during data collection. This data loss is more pronounced for higher echelon exercises because of the larger number of data packets involved. Even for platoon level exercises, if many data packets come in as a short burst, data loss might occur during the burst period.

To correct or minimize the adverse effect of data losses, the following measures have been taken as described below.

(1) Use a large buffer to buffer incoming data while the disk write is in progress.

(2) Write more than one data packet to the hard disk for each disk write to increase the throughput.

(3) Use Vehicle id's for data filtering.

(4) Periodically close data file.

Buffering

It is recognized that data packets are not coming in at a steady rate. There will be brief periods of time when there are plenty of incoming data packets while at other times the data rate is lower. Due to the fluctuating data rate, a large buffer can be used to buffer the packets at times when the hard disk is not able to keep up with the high data rate. When the data rate drops lower, the hard disk will be able to write the data packets from the buffer.

Writing Multiple Packets for Each Disk Access

The read-write head of the hard drive is a slow mechanical device. It needs to move itself to the right position on the hard disk before electronic transfer can take place. The average access time (to move the read-write head) ranges from 15 to 40 milliseconds while the transfer rate is roughly one to 10 Meg bits per second. For discussion purposes, assume 20 milliseconds for the access time and 5 Meg bits per sec for the transfer rate. Assume also that each data packet is 250 bytes in size. The time required to write a 250-byte packet to the hard disk will be:

access time + electronic transfer time (where access time = around 20 milliseconds and transfer time = 0.5 milliseconds).

Apparently, the electronic transfer is much faster than the disk access. It follows that it would be more efficient to transfer multiple packets for each disk write because each disk write involves only one disk access even though it transfers multiple packets. By saving the number of disk accesses, the throughput will improve considerably.

Vehicle IDs as Data Filter

UPAS currently allows the user to specify a list of vehicle id's for data filtering. Only packets with the matching vehicle id's are collected while other packet types are discarded. By selecting the vehicle id filter, the amount of data to be written to the disk can be minimized considerably and data loss is less apt to occur. Note that, to facilitate user input, UPAS has been modified to allow the specification of wildcards (*) for any of the three fields in a vehicle id. It also allows the specification of "0.0.0" for vehicle id. In SIMNET, some indirect fire weapons use "0.0.0" as their vehicle id.

Periodically Close Data File

The UPAS data collection module has been modified to periodically close and reopen the file that receives collected data. The frequency for closing and reopening the file is tunable and is currently set to close and reopen the file for every 500 disk writes. The overhead for closing and reopening the data file is relatively small compared to the benefit accrued from such practice. This is necessary to ensure that the data file has its data saved and closed properly at different points during data collection. If the data file is only closed at the end of a long data collection session and the system crashes before the end of the session, the data file would have been empty.

APPENDIX B
EXERCISE TIMELINE

The Exercise Timeline Display provides a graphic display to illustrate the times at which major events occur during an exercise for the vehicles in a platoon. The display of these major events in an easily comprehensible format will help the exercise participants to recall what have happened during the exercise when they are conducting the After Action Review. This will help them focus their attention on these critical points in time at which major events occur. Other AAR aids, such as Planview display or Battle Snapshot can then be taken to focus on these critical times.

The Exercise Timeline Display can be used for either the Blue or Red Force as selected by the user. The major events displayed in the Exercise Timeline can be grouped into three categories:

"Move" events
"Shoot" events
"Communication" events

Move Events

The Timeline shows the time interval (shown as a bar in the display) between the first and last (non-disabled) vehicles in the platoon crossing any one of the control measures; Assembly Areas, Check Points, Objectives, Phase Lines, Lines of Departure, Starting Points, Release Points. Note that disabled vehicles are not included when calculating the time interval. For Starting Points, Check points, Release points, vehicles coming close to within 50 meters of the said points are considered crossing. For Phase lines, Lines of Departure, vehicles need to actually cross the said lines to be considered crossing. For Assembly Areas, vehicles are considered to have left the areas when they are more than 100 meters from the area. For Objectives, vehicles are considered to have reached the Objectives when they are within 100 meters of the said Objectives.

The Timeline also shows the times during which no vehicles in the platoon moved. Disabled vehicles are excluded from the calculation.

Fire Events

The Timeline shows when artillery fire impacts near a unit. Any artillery or mortar fire within 500 meters of any of the vehicles in the platoon is considered near.

The Timeline shows when enemy fire is first received by a specific platoon. The firing event is defined in terms of the location of the impact relative to the location of the platoon. More specifically, a platoon is considered to have received fire if a round impacts within 500 meters of any vehicle in the platoon. This computation does not consider whether the firing event results in a miss, hit, or kill.

The first friendly fire delivered by the platoon refers to the first firing event in the exercise in which one of the platoon's vehicles is the firer. This calculation does not consider the result of the firing event or whether the round impacts near the enemy. If a vehicle should accidently fire while the gun tube is pointed opposite the direction of the enemy, this firing event would still be identified as the first firing event for the platoon.

The Timeline also shows each time that a friendly or enemy vehicle is destroyed. This calculation considers only those firing hits resulting in a kill, rather than a hit.

The shoot events in the Exercise Timeline are color-coded as follows: "First friendly fire delivered", "First enemy fire received", "Artillery fire near unit" are color-coded based on the force of the firing vehicles (blue for Blue Force and red for Red Force). "Enemy vehicles destroyed", "Friendly vehicles destroyed" are color-coded based on the force of the vehicles destroyed.

Communication Events

The communication lines shows when radio communications occurs and shows the type of communication (Report, Call for Fire, Order, Request for Information, or Miscellaneous). For a platoon-level Timeline, the communications are on the platoon net. For a company-level Timeline, the communications are on the company net. These communications data must be collected by a trainer or researcher monitoring the appropriate net. The data must be loaded into a UPAS relational database table with column definitions as shown in Table B-1.

An Exercise Timeline is shown in Figure B-1. This is for a higher echelon display in which more than one platoon is involved. Each display page can accommodate two platoons. For a company with more than two platoons, it might take more than one page to display the Exercise Timeline for all the platoons in the company. The user can page through all the pages by pressing the appropriate function keys. At the platoon level, only one page is sufficie to accommodate the display for the platoon.

Table B-1.

Communications Table.

Field Name	Manner Coded	Description
ttime	Time	Approximate time that communication occurred.
side	1 character.	R for REDFOR and B for BLUFOR
orgcode	3 characters.	Three digit code used to indicate the platoon or company sending the message. Companies are coded as follows: 100 for Company A, 200 for Company B, 300 for Company C, and 400 for Company D). Platoons are coded in the last of the three digits (1 for 1st platoon, 2 for 2nd platoon, 3 for 3rd platoon, 4 for 4th platoon, 5 for attached platoon, and 6 for Company HQ). For example, the 3rd platoon of Company A would be 103.
commotype	1 character	This is a one character code used to indicate the message type, as follows: 1 for a report; 2 for an order; 3 for a call for fire; 4 for a request for information; and 5 for miscellaneous.

The length of the time scale covers the length of the exercise. The starting time is marked on the time scale in the format of "hhmm" as the first label on the time scale. The other time labels to be marked on the time scale will be in the format of "mm" without the preceding "hh" to avoid a cluttered appearance. The precision of each time label chosen depends on the duration of the exercise. The shorter the exercise, the finer (more precise) will be the time label.

APPENDIX C
LINE-OF-SIGHT DISPLAYS

During after action reviews, it is often necessary to know why two hostile units did not fire at each other even when they were relatively close to each other. It might be that the units cannot see each other because their line-of-sight (LOS) was blocked by intervening terrain. Therefore, it is necessary to have a tool to help users decide if LOS is blocked.

The Line of Sight Display capability has been incorporated into the Battle Snapshot module in UPAS. The algorithm used in LOS determination in UPAS considers the intervening terrain and some terrain features including treelines and tree canopies. It does not consider other factors such as weather (foggy or clear sky) and time of day (under bright sun light or in a dark night) that might also influence the visibility of the observer. These other factors are not played in SIMNET.

Description of LCS Algorithm

The algorithm used in LOS calculation was developed by another IST project which uses the Borland C development environment. The algorithm was adapted to be used in UPAS which uses the Microsoft development environment. For detailed description of the algorithm used in LOS calculation, please refer to Petty and Campbell (1992). Only a brief description of the algorithm is presented here. Since the LOS calculation requires intimate knowledge of the terrain database, the terrain representation is first described. The algorithm used in LOS calculation is then presented.

Terrain Representation

UPAS uses a terrain database compatible with SIMNET. In the database, the land of the terrain is represented by a large set of polygons. Any two polygons might be next to each other in that they share some edges and vertices. The polygons are not necessarily on the same plane because of the 3-dimensional nature of the terrain (The terrain rises and falls in height as it goes across hills and valleys).

Besides land polygons, there are treelines and tree canopies. Treelines are rows of trees; each represented by a line segment with a constant height. Tree canopies are groups of trees represented by treelines outlining the borders of the canopies and sets of polygons forming the covers for the canopies. The land, treelines and tree canopies are the terrain features that are used in LOS calculations. Other features like rivers, roads, building structures are not considered in the calculation.

All the polygons, and edges and vertices that form the polygons are organized in the database in a manner that facilitates their retrieval by means of (patch, grid) pairs. The terrain as projected on the x-y plane can be divided into 500m x 500m square patches and each patch is further subdivided into sixteen 125m x 125m square grids. Each patch is identified by a patch index and each grid is identified by a grid index. Each polygon, edge and vertex can be associated with one or more (patch, grid) pair. The database is organized such that given a (patch, grid) pair, all the edges of polygons (for land and tree canopies) and of treelines reside within the patch and grid can be easily calculated and retrieved.

LOS Calculation

The LOS calculation involves the following steps:

(1) Given an observer and a target, the line connecting the two points (LOS) is projected to the x-y plane. The Bresenham's algorithm is then used to locate all the (patch, grid) pairs that can be traversed by LOS.

(2) For each (patch, grid) pair found in step 1, determine all the edges that are covered by the (patch, grid) pair. Only these edges will be used for LOS calculations.

(3) For each edge located by step 2, determine if the LOS is blocked by the edge. This is determined as follows. First, the point of intersection of the edge and LOS on the x-y plane is calculated. Then the height of the edge and LOS at the point of intersection is compared. If the edge is higher than LOS at the point of intersection, the LOS is blocked.

Note that more than one edge might block the LOS. In UPAS, the edge nearest to the observer and blocking the LOS is identified and a green line is drawn from the observer to the blocking edge and a red line drawn from the blocking edge to the target. If no blocking occurs, the entire line from the observer to the target is colored green.

Because of the amount of computations involved in LOS calculations, it is highly recommended that a high-powered computer, such as a 386 or a 486 PC, be used to speed up the calculations.